Grid Prototype Project (caGRID?)

June 24, 2004

Agenda

- Introduction
  - Team Members
    - Manav Kher – (SAIC)
    - William Sanchez – (SAIC)
    - Brian Gilman – (Panther Informatics)
    - Steve Pouros – (Panther Informatics)
- Requirement analysis
- Technology evaluation
- caGRID framework
- Prototype demo

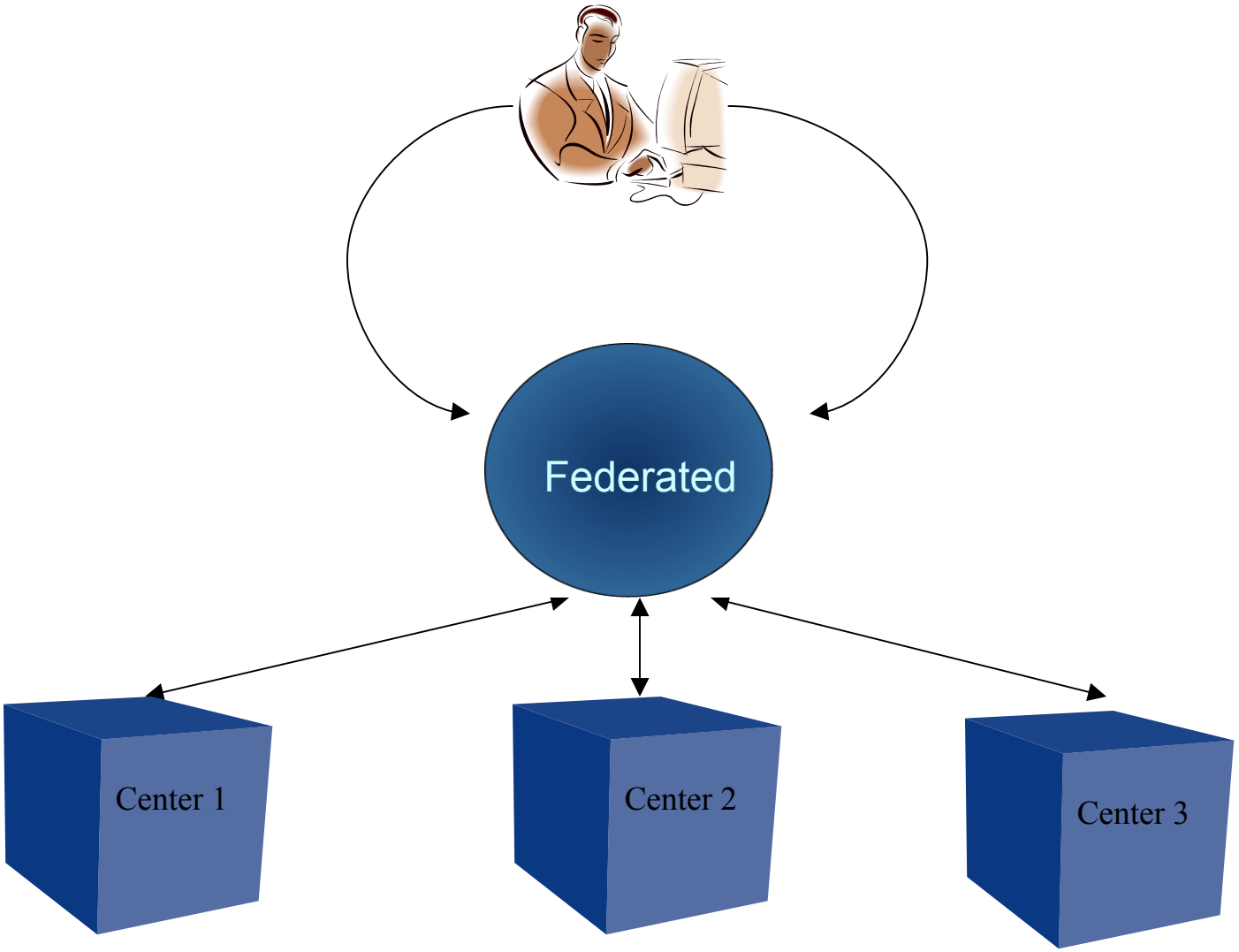
Problem Statement

- Production of data is outstripping our ability to analyze it
- The research community may not be aware of other work and datasets
- Researchers may not tag data with the definition of the data they produce
- Semantic information is not often encoded nor included with data sets
- “Data Islands” or “Silo’s” of information are produced based on the problems outlined above
  - A small group of knowledgeable people transmit data amongst themselves
- “Modern” exploratory research requires the integration of disparate databases of biological information to explain results
- To elucidate the mechanism behind disease we must aggregate data from many databases

Vision

- Provide the “Grid For Cancer Research” so that we may:
  - Raise awareness of disparate datasets in the biological research community
  - Allow research groups to exchange datasets with ease
  - Allow research groups to understand the semantics of the datasets that they publish without always having to get on the phone
  - Allow for quicker publication of the analysis of integrated data

The Grid Concept



What Criteria Do We Use To Assess The Technology?

- The caGRID project team utilized the following metrics to assess the applicability of grid technologies to biological problems
  - Ability to handle large datasets
  - Robust protocols for data transmission
    - Quality of service (QOS)
  - Protocols for meta-data and semantic representations of datasets
  - Robust publication, search, and query capability
  - Mature application programming interfaces (APIs)
  - Maturity of technology
    - Change in codebase and API
    - Quality assurance best practices in place
    - Change management system in place with release notes and backward compatibility taken into account
    - Ease of installation and administration (what is the burden on system administrators/users to adopt technology)
    - Acceptance in the engineering community

What Technologies Exist To Produce a "Bio-Grid" (caGRID)

- Web Services
  - Bio-MOBY
- Grid Services
  - OGSA-DAI
  - Globus
  - SRB
- P2P
  - JXTA
- Hybrid

  - Chinook (Web services and JXTA)
- Home Grown

Related projects, technologies and standards.

OGSA-DAI
(Data grid)

(Data grid project)

Jena2
(Semantics)

SDSC
SRB
Storage Resource Broker
(Data grid application)

(Grid project)

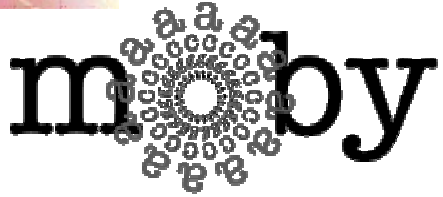
(Grid infrastructure framework)

The North Carolina BioGrid Project

Web Services

JXTA
(P2P technology)

(Data grid application)



(Web service registry for Bioinformatics)

(Grid project)

What is needed to make the Grid “Bio-Accessible”?

- Submit caGRID prototype project to the caBIG - Architecture Workspace for evaluation
- Increase understanding of grid computing in the biological sciences - caGRID project
- Produce a “best practices” and “lessons learned” document/presentation for the community to learn from our experiences
- Provide a forum for users to explain their requirements and pain points

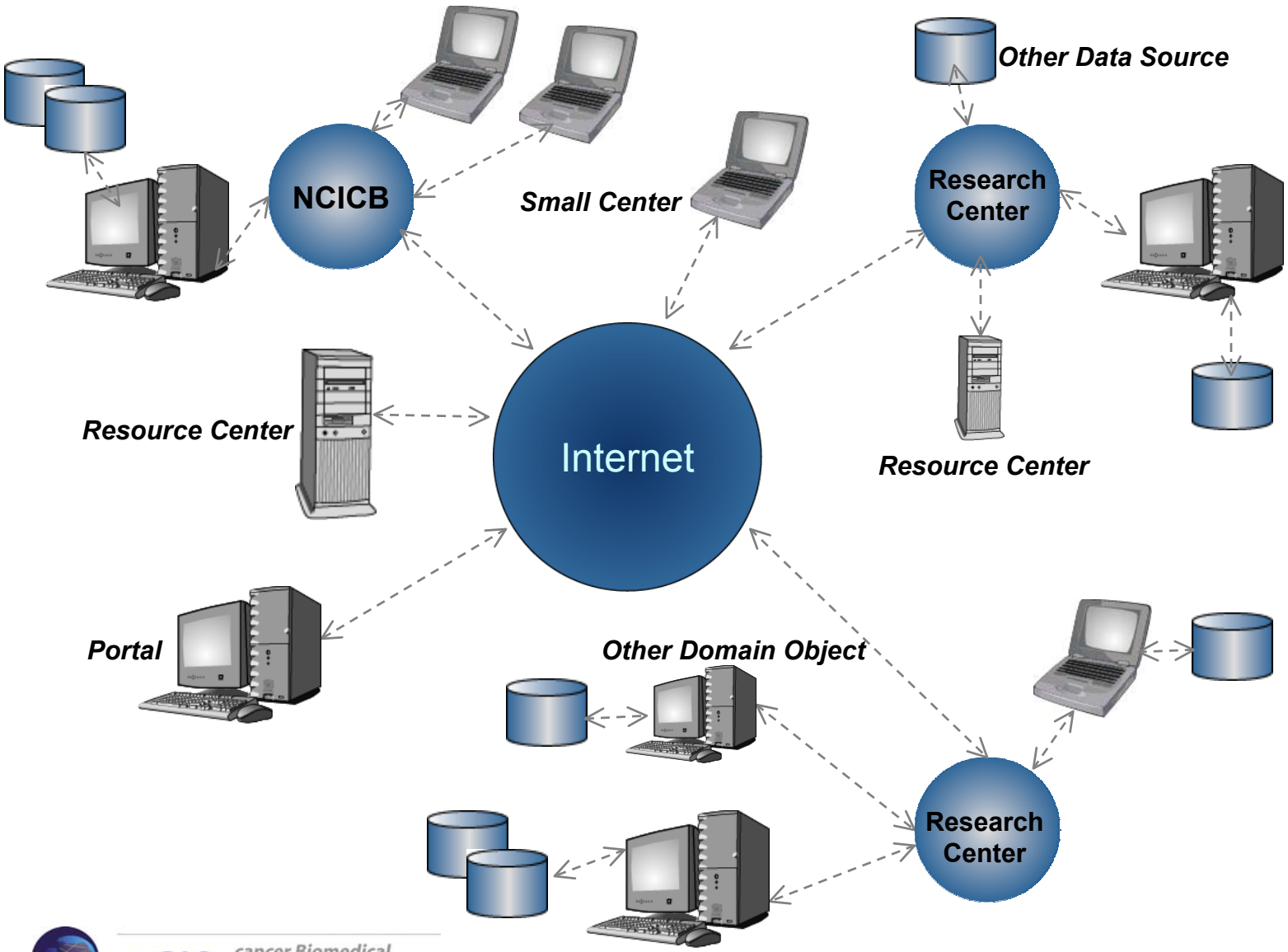
How Do We Realize a Bio-Grid?

- Provide Semantic representations and intuitive query of Datastores & Services
  - “Computer, find me all clinical trials that are in phase I and are studying neoplasia” vs. “Select * from clinical_trial_data ctd, disease d where ctd.id = d.id and d.desc like ‘%neoplasia%’
  - Find me all genes and their aliases that are associated with neoplasia
  - Find me all micro-array experiments that have been used in studying neoplasia
  - Find me all pathways that have been implicated in neoplasia
  - Put together a compelling, publishable, “drugable” story based on these queries

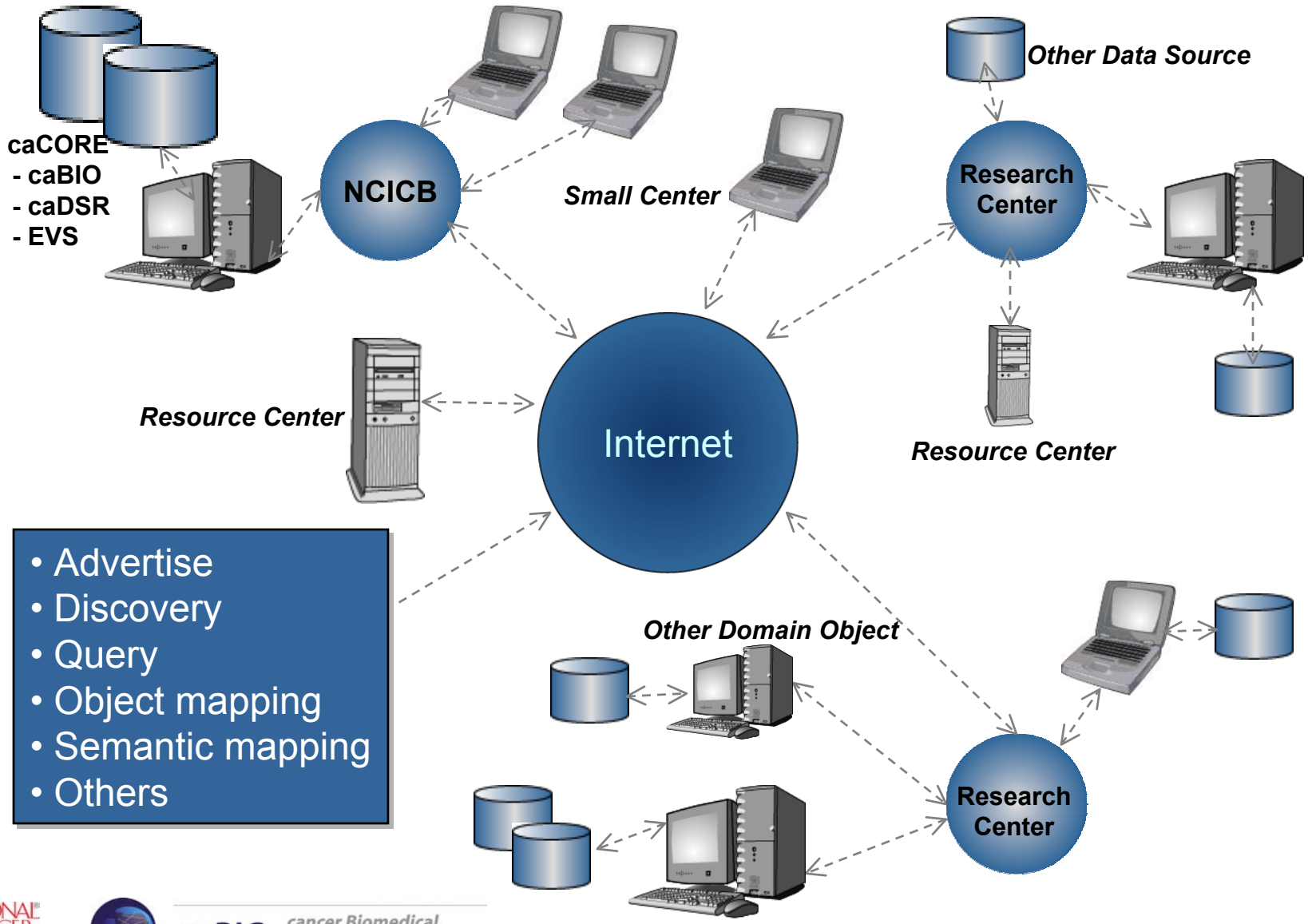
What Did We Choose And Why?

- Open Grid Services Architecture – Data Access & Integration **(OGSA-DAI)**
  - Based on Globus toolkit
  - Large development community
  - Written on top of web services technology
  - Best documentation with updated documents provided upon every release
  - Provides best tradeoff of complexity for ease of development
    - Addition of features is relatively straight forward

Grid requirements

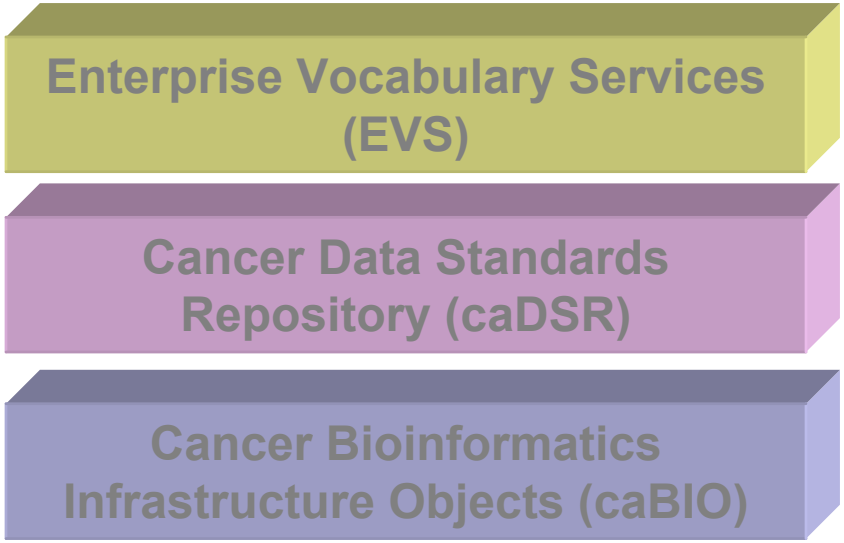


Grid requirements

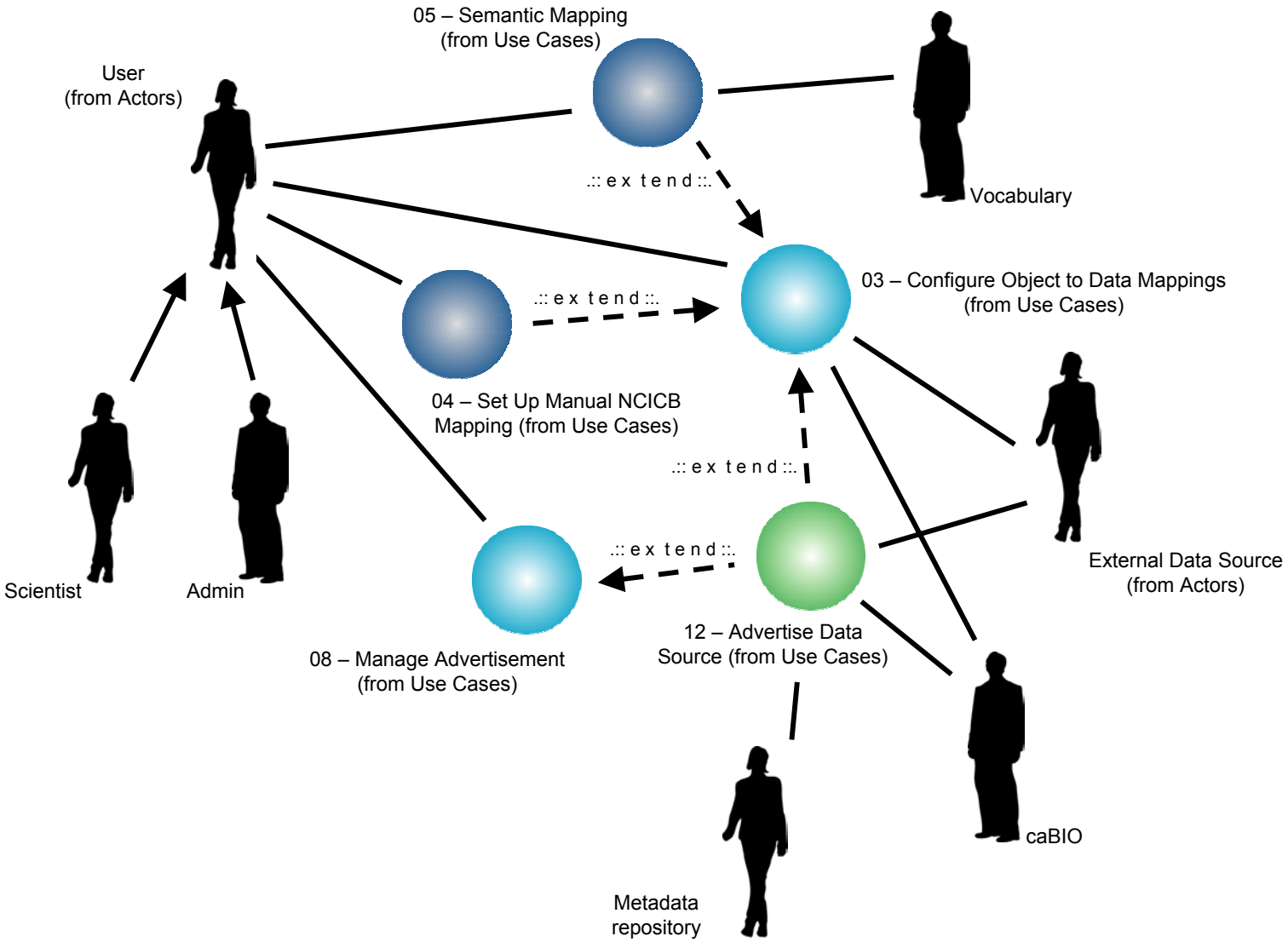


Prototype based on caCORE

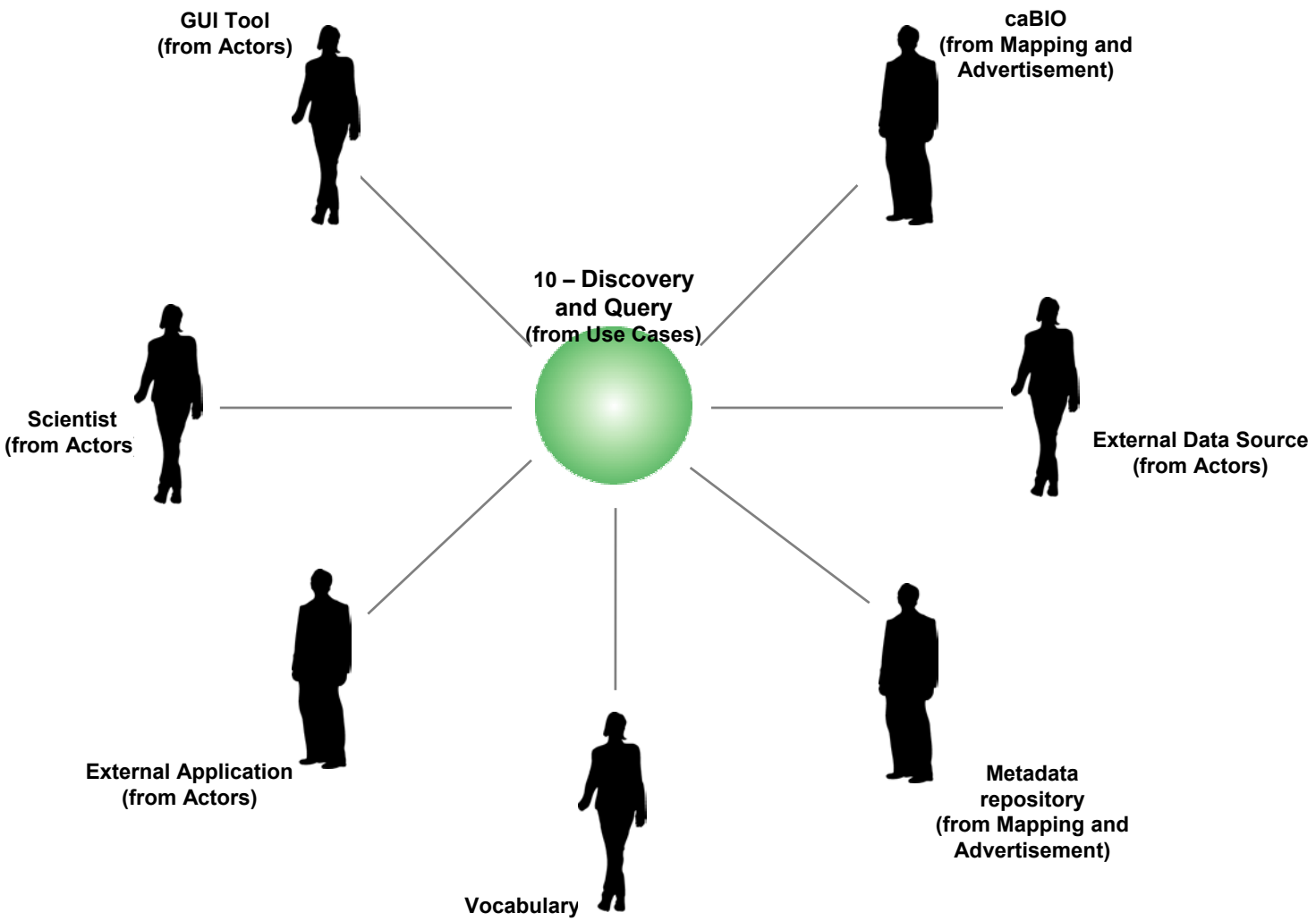
- **ca**ncer Common Ontologic Representation Environment (caCORE)
- caCORE is the technology stack that facilitates data integration across multiple scientific disciplines



Advertisement and Mapping



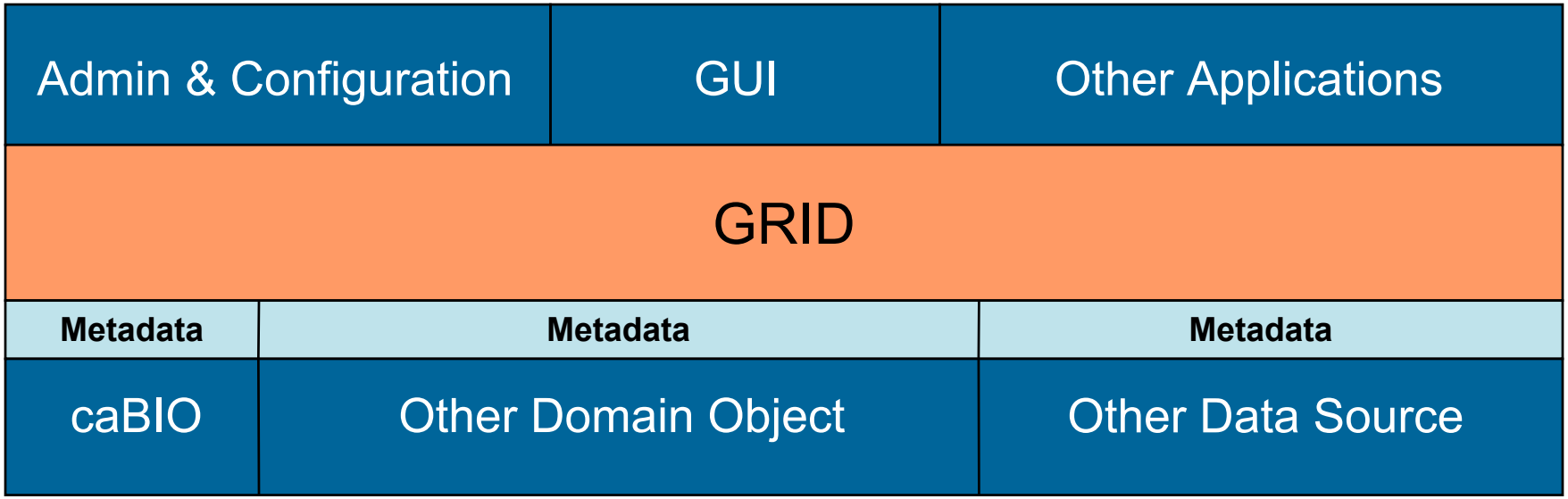
Discovery and Query



Requirement Prioritization

	Prototype
Goals	-Query Data and Discovery Service – Full -Advertise services – Command line -Startup – Command line -Shutdown – Command line -Install – Basic
Requirement	-Define and prioritize requirements -Perform technology evaluation (Grid, Semantics) -Define architecture -Implement prototype – caBIO -Semantics - service level -Test GRID technology/framework

Preliminary Architecture



Significant Use Case vs. Candidate Technologies

	Advertise	Discovery	Query	Object Mapping	Semantic Mapping
Web services	-UDDI -Extend UDDI. -Create a WS / server code.	-UDDI -Extend UDDI. -Create a WS / Server code.	- caBIO web services.		
Globus, OGSA-DAI, DQP	-Instantiate a grid service. -Registry new service. -Notification -Metadata Framework	-Indexing services (service data providers, data aggregators, grid service registry).	-Object model / caBIO java api. -Data bases (RDB, XML)		
SRB	- MCAP / Metadata service.	- MCAT / Metadata service.	- SRB server / FS, DB, Obj.		
Jena2	- Improve service description.	- Improve service discovery.			-RWU Ontology languages. -Representation of semantic obj.
MCS / caDSR	- Metadata service	- Metadata service			
- OJB				Customize xml representation to model other DB.	

OGSA and Globus Toolkit 3

- **OGSA:** Open standard architecture for next generation grid-services to enable the creation, maintenance, and integration of grid services maintained by virtual organizations.
- **OGSI:** A core component of OGSA, which provides a uniform way to describe grid services and defines a common pattern of behavior for all grid services. In short, this architecture defines grid service behaviors, service description mechanism, and protocol binding information by using web services as the technology enabler.
- **Virtual Organizations:** Coordinated resource sharing and problem solving in dynamic, multi-institutional organizations. sharing is with direct access to computers, software, data, and other resources by a range of collaborative problem solving and resource brokering strategies. The sharing is controlled with resource providers and consumers defining what is shared, who is allowed to share and the conditions under which sharing occurs. The set of individuals and/or institutions defined by the sharing rules from the virtual organization.
- **GT3:** The Globus software technology toolkit version 3 is the major reference implementation of the OGSI standard.

OGSA-DAI

The OGSA-DAI project is concerned with constructing middleware to assist with access and integration of data from separate data sources via the grid. It is engaged in identifying the requirements, designing solutions and delivering software that will meet this purpose. The project was conceived by the **UK Database Task Force** and is working closely with the Global Grid Forum **DAIS-WG** and the **Globus Team**.

OGSA-DAI is a UK **DTI Funded** e-Science Grid Core Project involving the following partners:

National e-Science Centre

ESNW

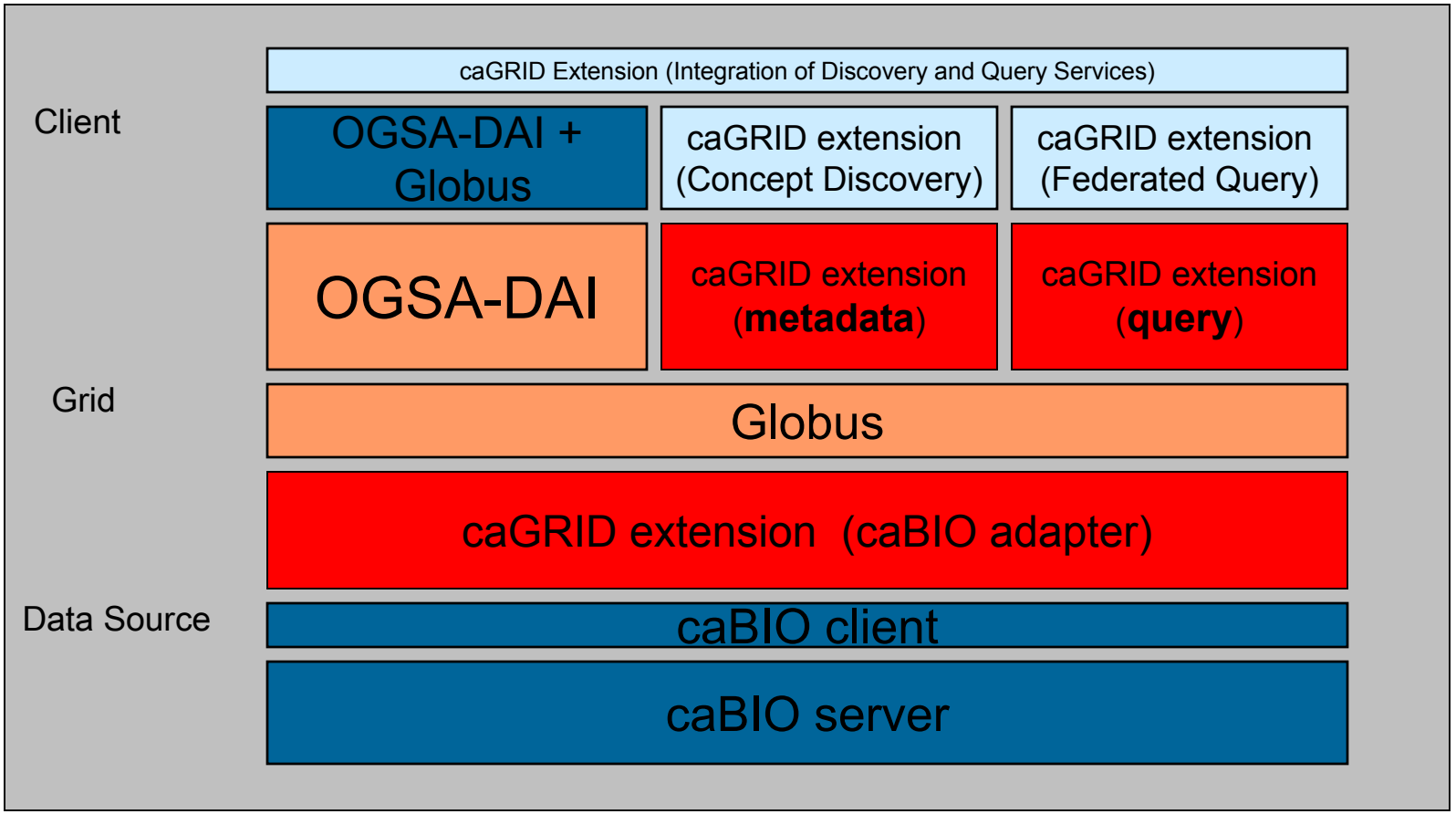
neresc

epcc

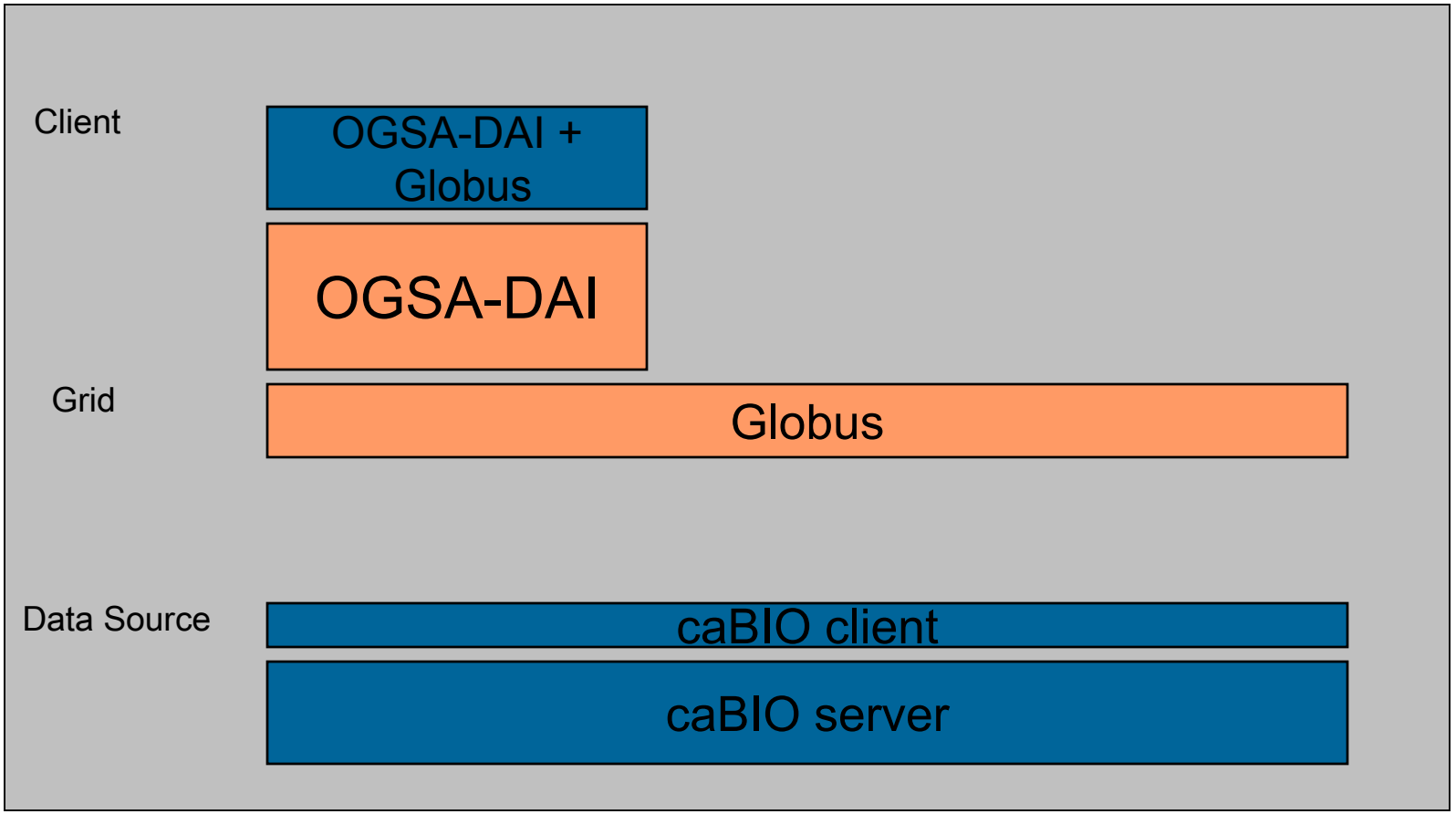
ORACLE

IBM

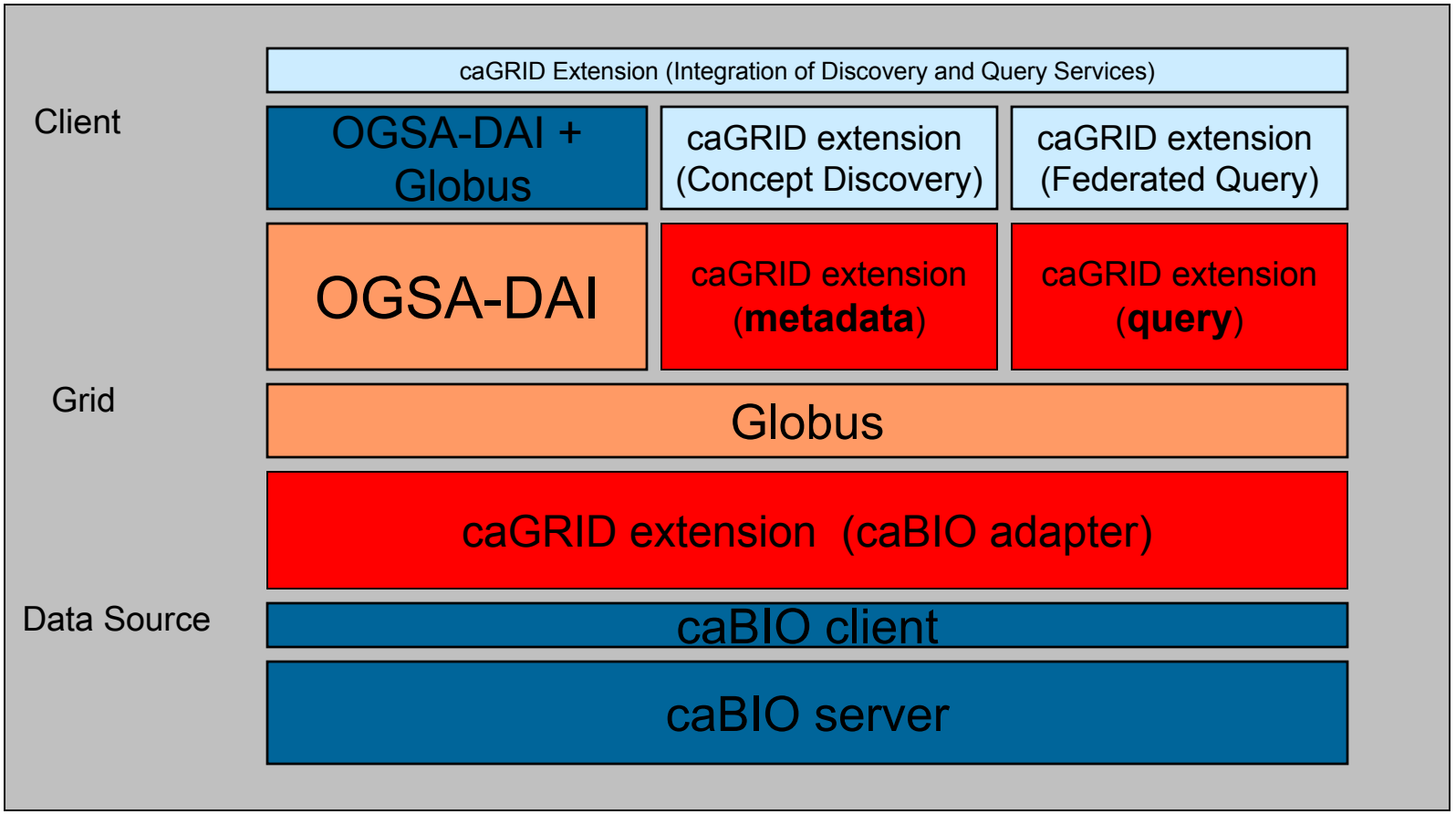
caGRID Core architecture



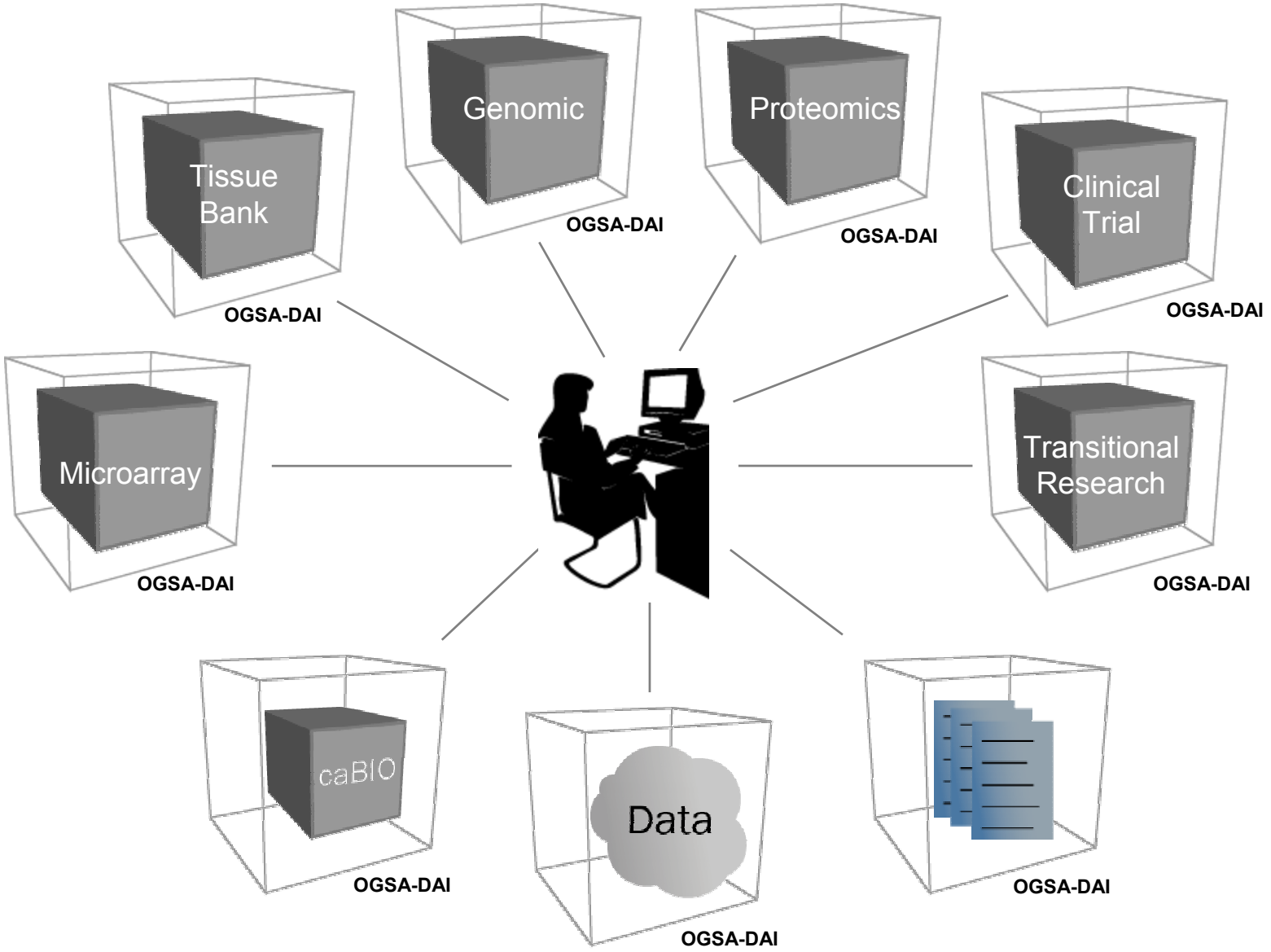
caGRID Core architecture



caGRID Core architecture

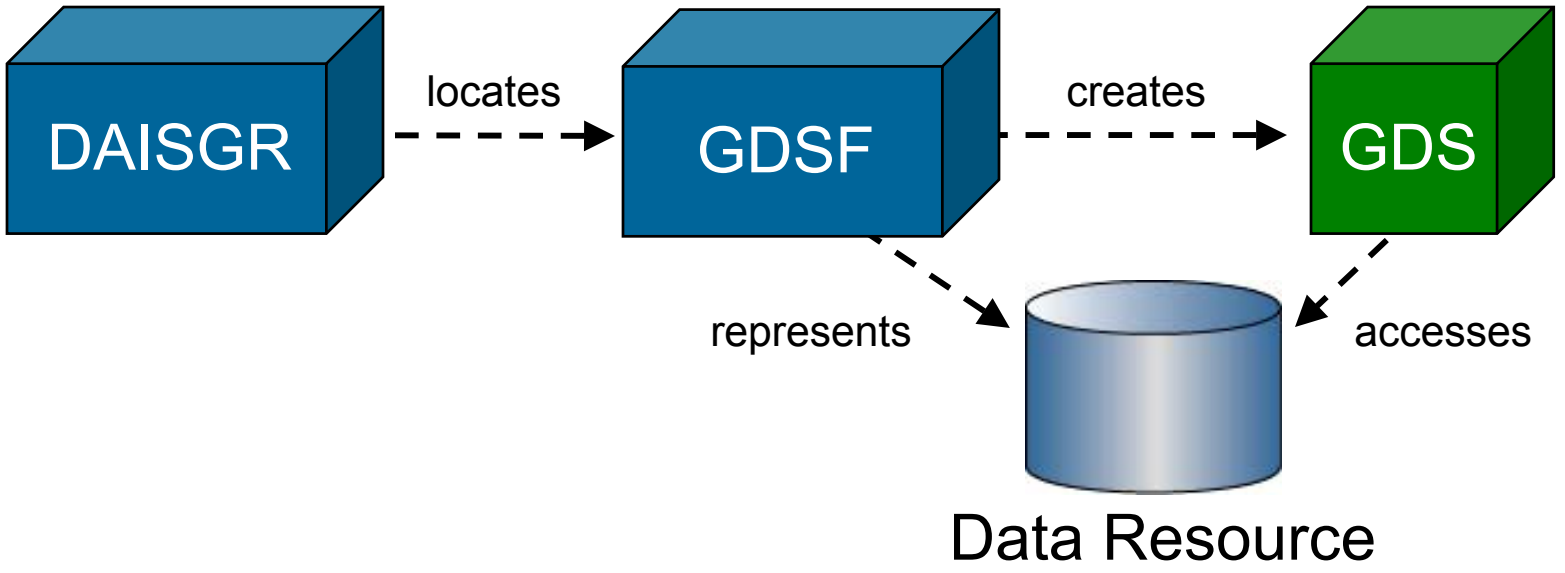


Facilitating data access



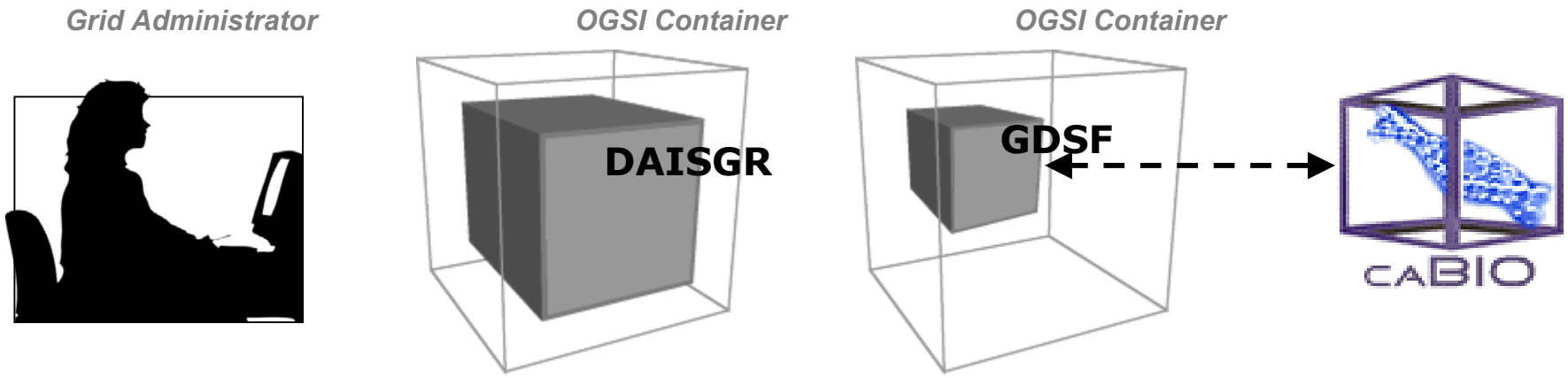
OGSA-DAI Services

- OGSA-DAI uses three main service types
  - **D**ata **A**ccess & **I**ntegration **S**ervice **G**roup **R**egistry (DAISGR) for discovery
  - **G**rid **D**ata **S**ervice **F**actory (GDSF) to represent a data resource
  - **G**rid **D**ata **S**ervice (GDS) to access a data resource



Interaction Model: Start up

1. Start OGSI containers with persistent services.
2. Here GDSF represents caBIO database.



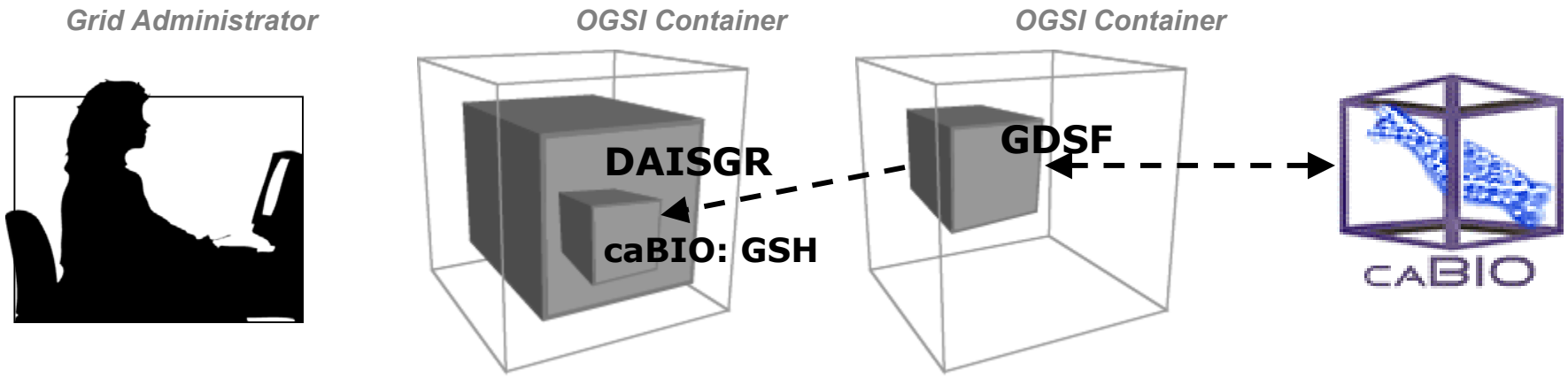
-DAISGR (registry) for discovery

-GDSF (factory) to represent a data resource

-GDS (data service) to access a data resource

Interaction Model: Registration

3. GDSF registers with DIASGR



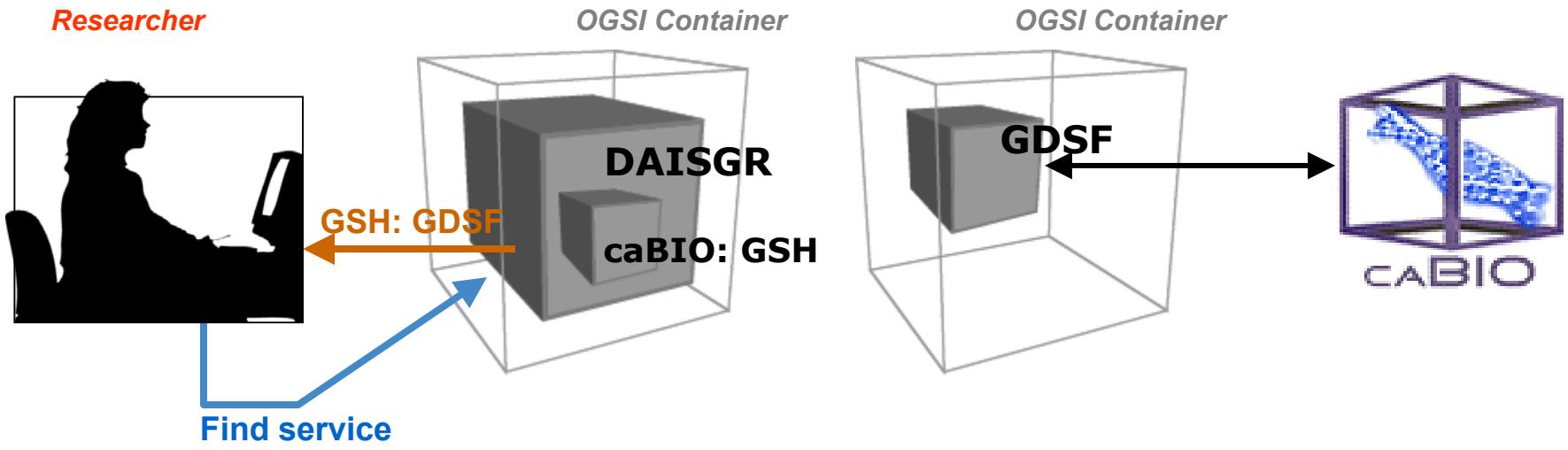
-DAISGR (registry) for discovery

-GDSF (factory) to represent a data resource

-GDS (data service) to access a data resource

Interaction Model: Discovery

4. Client wants to know about caBIO. caBIO can…
   (i) Query the GDSF directly if known or
   (ii) Identify suitable GDSF through DAISGR.



-DAISGR (registry) for discovery

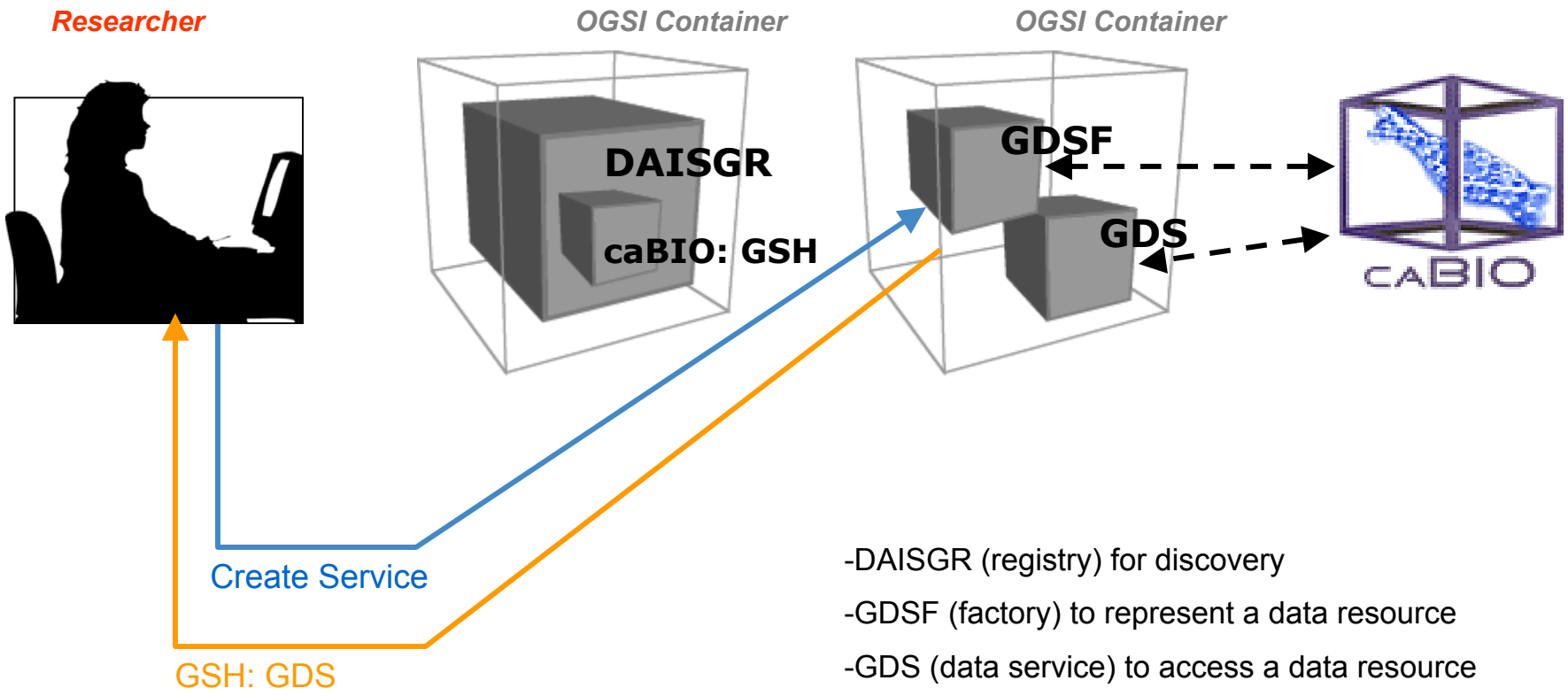
-GDSF (factory) to represent a data resource

-GDS (data service) to access a data resource

-Grid Service Handler (GSH)

Interaction Model: Service Creation

5. Having identified a suitable GDSF client asks a GDS to be created.



-DAISGR (registry) for discovery

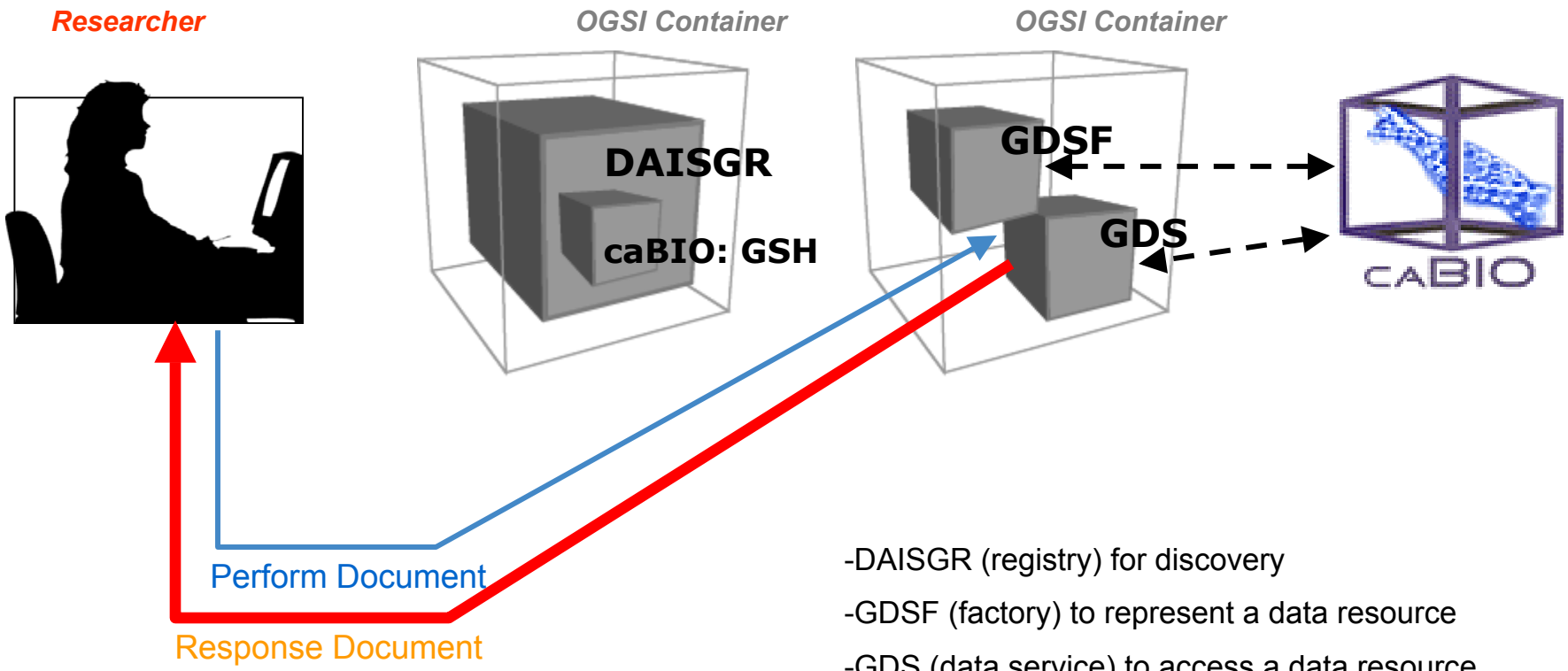
-GDSF (factory) to represent a data resource

-GDS (data service) to access a data resource

-Grid Service Handler (GSH)

Interaction Model: Perform

6. Client interacts with GDS by sending Perform documents.
7. GDS responds with a Response document.
8. Client may terminate GDS when finished or let it die naturally.



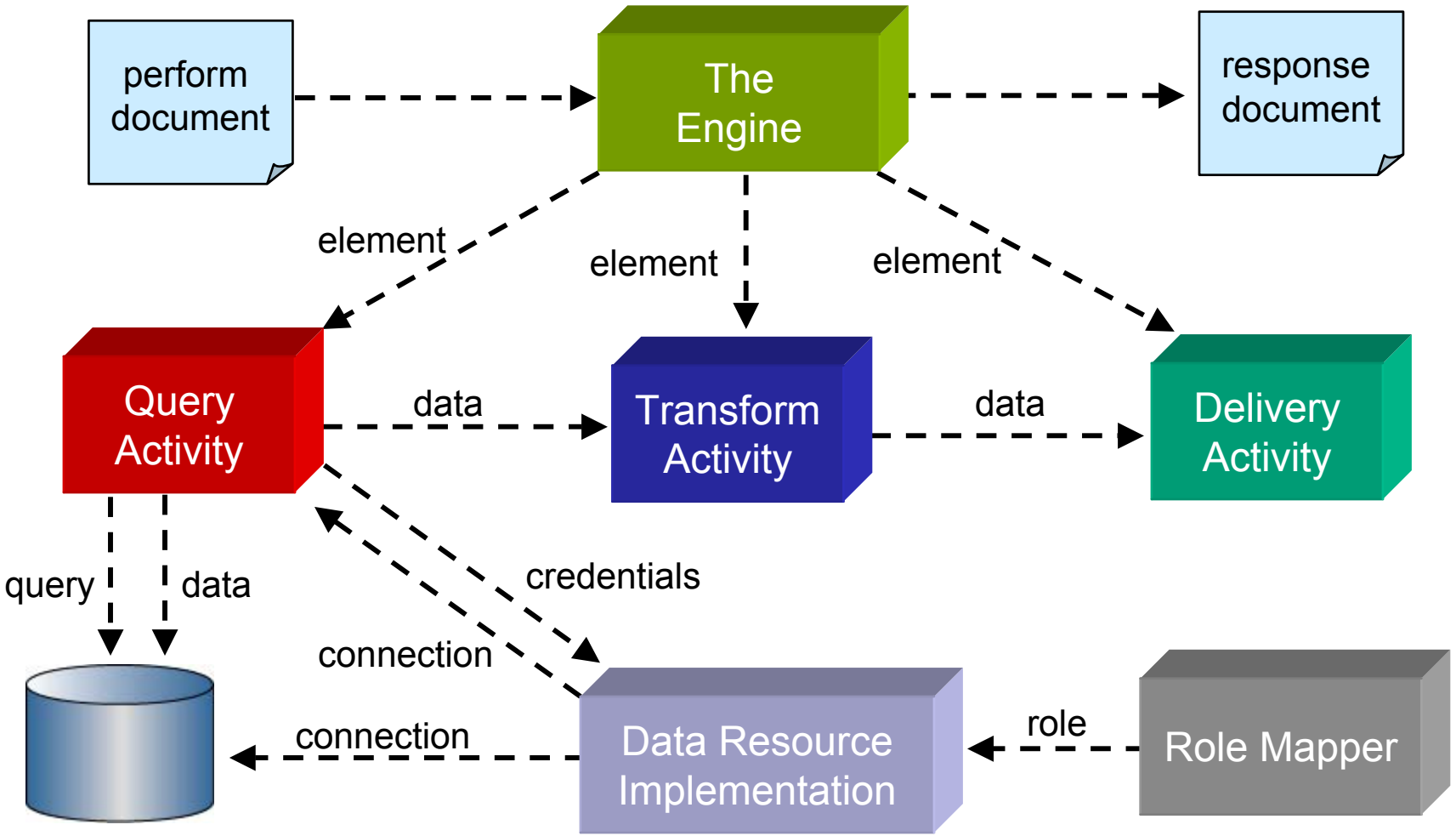
-DAISGR (registry) for discovery

-GDSF (factory) to represent a data resource

-GDS (data service) to access a data resource

-Grid Service Handler (GSH)

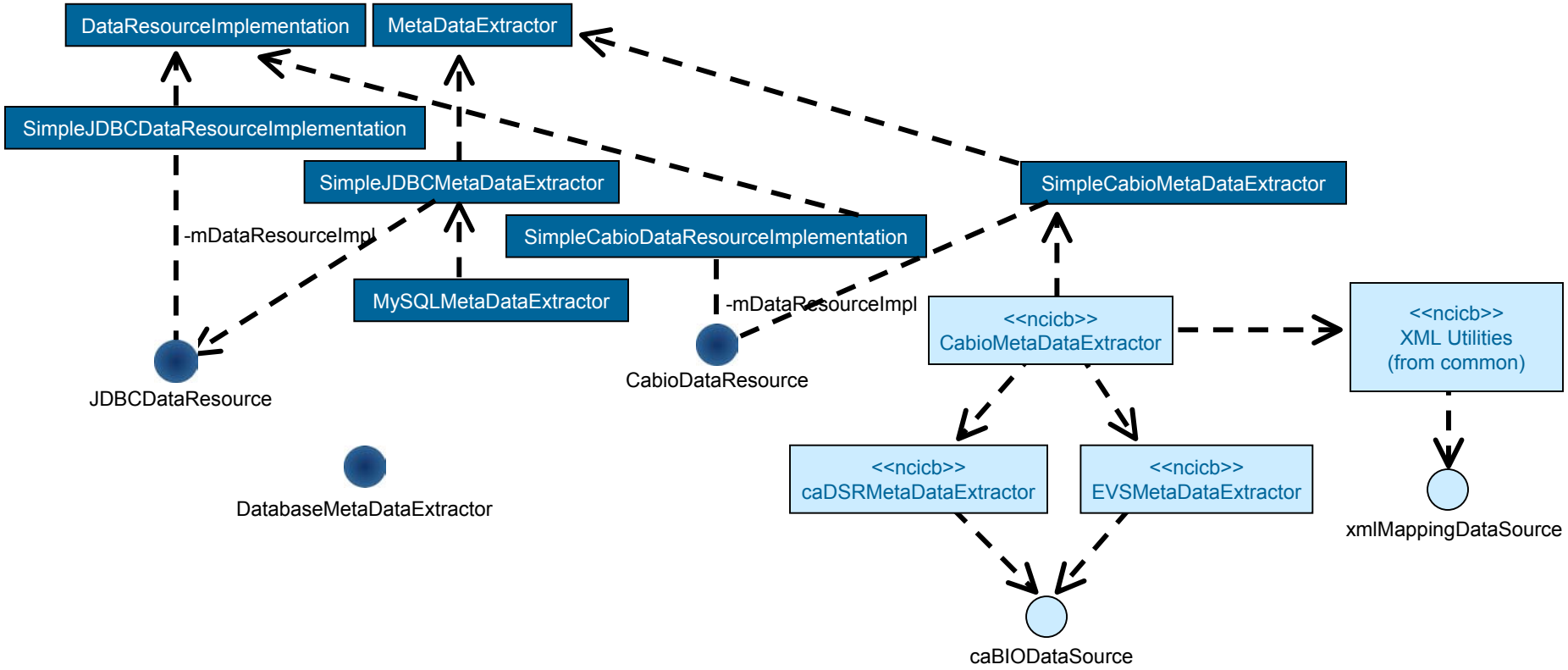
GDS Internals



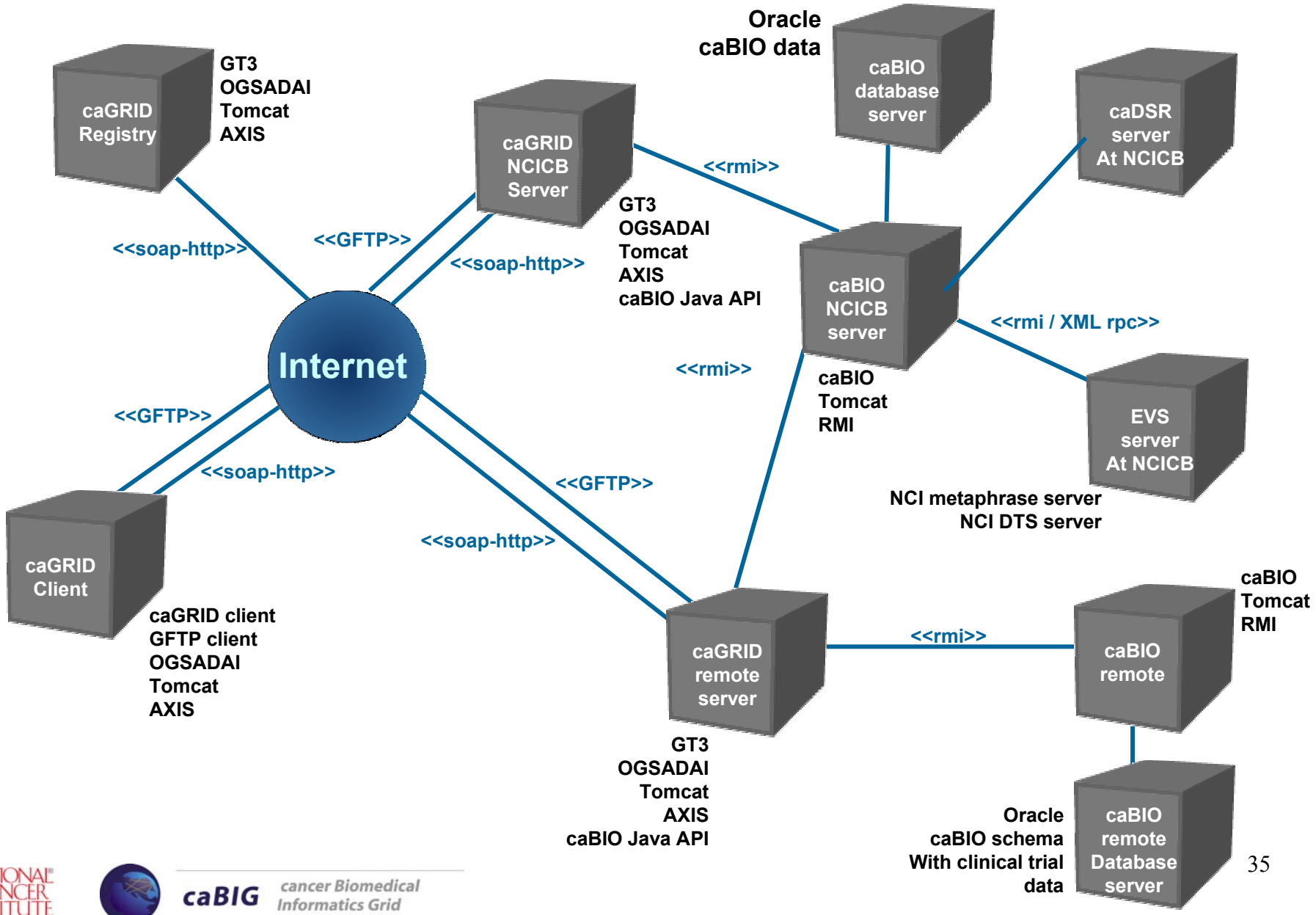
OGSA-DAI Extensions

- **New data source:** caBIO added as a new data source to the OGSA-DAI framework.
- **Activities:** Two activities that implements caBIO Java api.
- **Local caDSR:** Local xml representation of caDSR.
- **Query language:** Prototype query language with XML representation to query caBIO data source.
- **Metadata:** 2 new metadata categories added (caBIOSchema and ResearchCenterInformation). Includes the caDSR/EVS metadata extractor.
- **Concept Discovery:** Client extended to discover by concept name in the grid. The result set is the factories grouped by relationships.
- **Federated query:** Federated query by relationship (from discovery process). Send serialized queries to the factories that have advertised concept-relationship.
- **Sample Implementation:** A command line application that integrates Discovery and Query extension. The application help the user to perform discovery and federated queries.
- **Result set cached (Alpha):** The result set from a federated query will be store in a client xml database. Using another factory the result set can be query using xpath expressions.
- **Metadata query (Alpha):** ability to query metadata without using xpath expression.

Datasource – GDSF – Framework extension



Deployment Diagram (Prototype)



Prototype Demo

- Globus GUI – Grid service concept,
  - Registry
  - Metadata
  - Metadata Query
  - caBIOConfig
- OGSA-DAI GUI
  - Instantiate a factory
  - Submit perform documents
  - Show perform Documents
- Client and Client extensions
  - Add/Remove services
  - Discovery process
    - Clinical trial protocol, Diseases
  - Query services
    - Show me all phase I Clinical Trial Protocols
    - Show me all Trials for Neoplasia
    - Show me all Diseases that are associated with Phase I trials.
  - Sample using the caBIG middleware.

Demo starts here …

Lesson Learned

- There is an inherent learning curve in implementing grid technologies
- Grid technologies are still maturing and preparation for frequent upgrades is essential
- Common meta data structure and terminology is necessary to effectively describe services and data
- A common query language is important to support federated queries

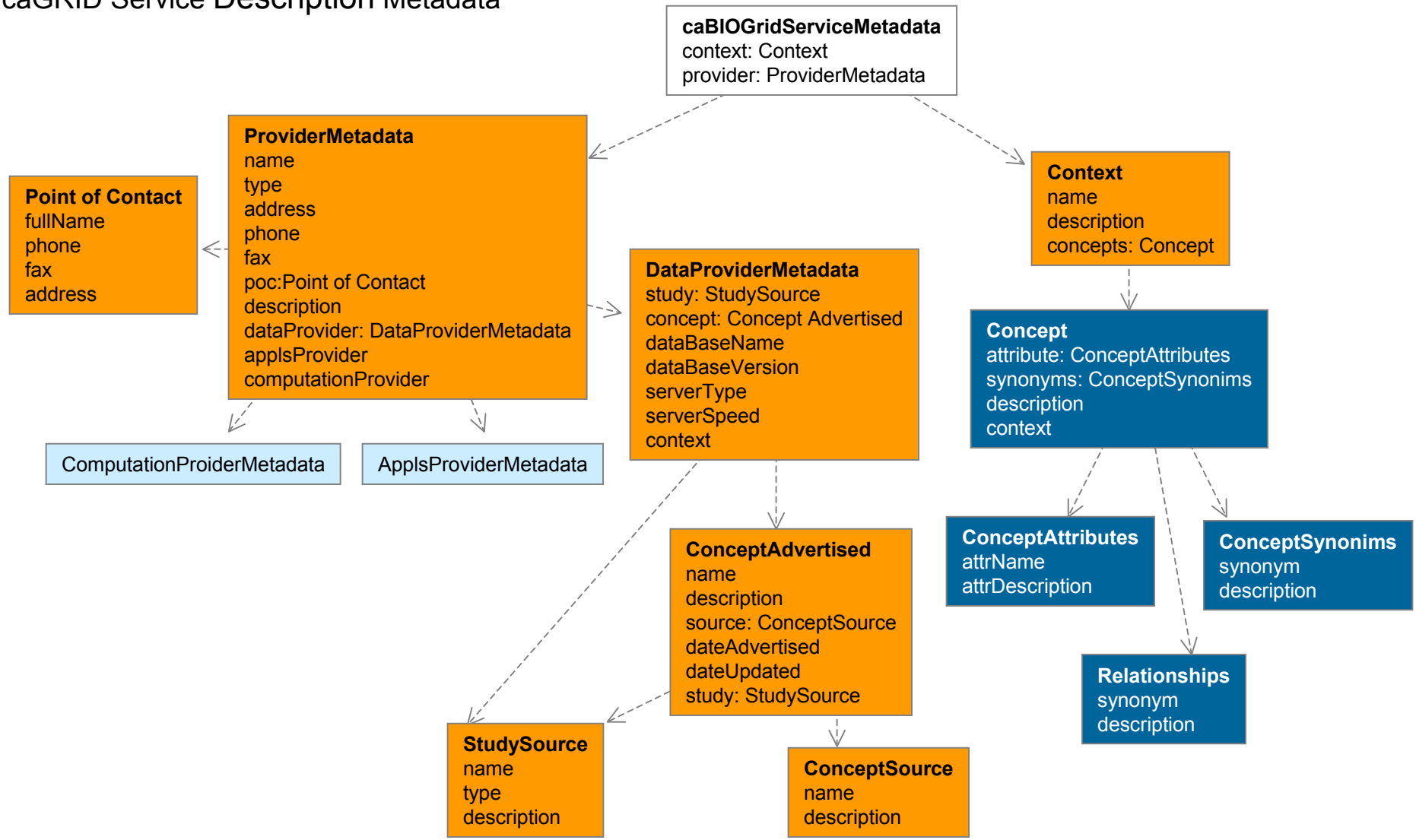
Future Recommendations

- Map existing use cases to caBIG needs
- Continue extending caGRID to produce a robust platform. Extensions include:
  - Upgrades to the latest version of Globus in support of web services standards
  - Design and implementation of user friendly GUI query tools
  - Implementation of additional grid services including
    - Security
    - File Sharing
  - Enhance strategy for semantic web interoperability
- Document caGRID efforts
  - Draft a white paper describing evaluation and prototype efforts
  - Draft developer guides to assist in establishing grid connections

Q&A

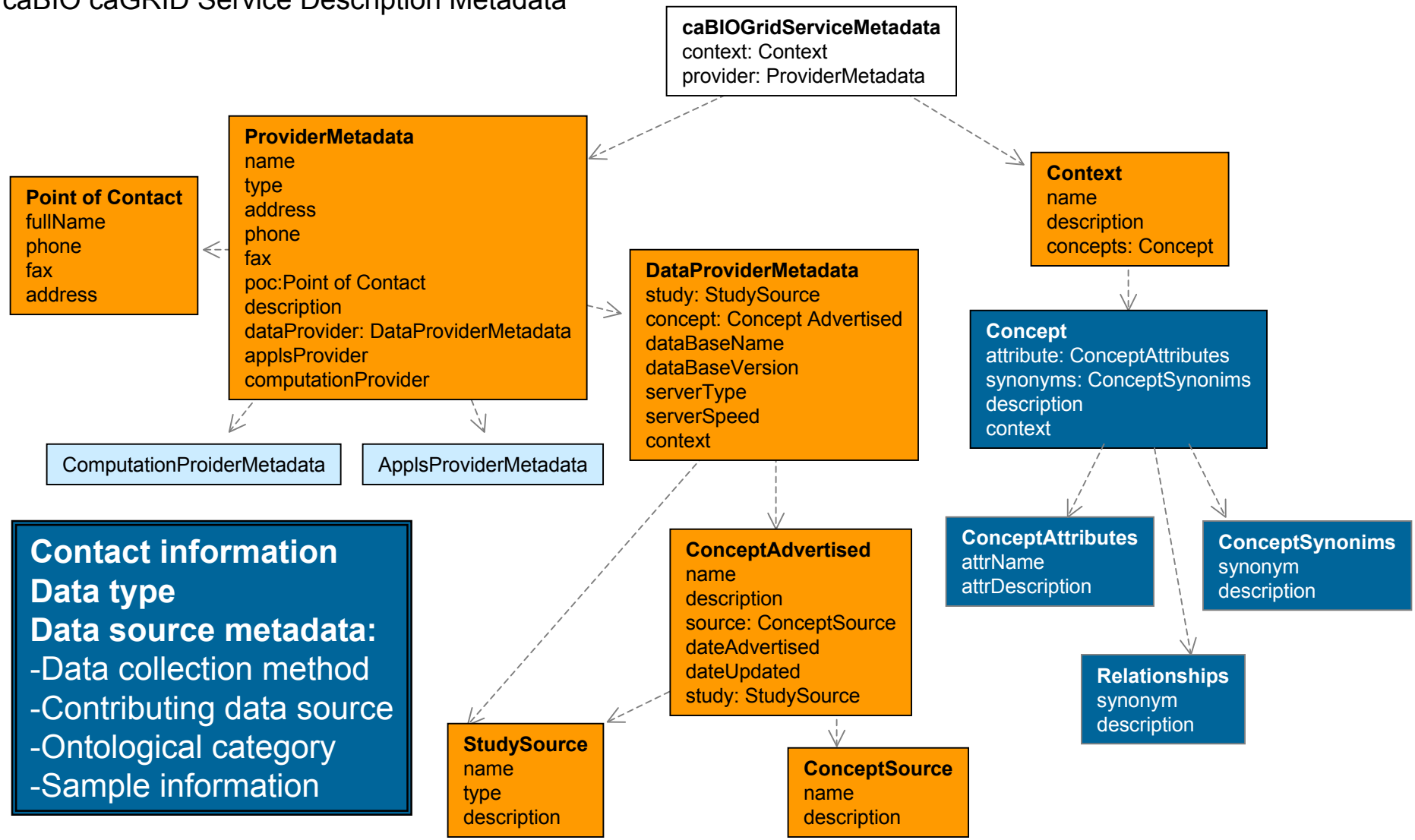
Metadata - GDSF

caGRID Service Description Metadata



Metadata - GDSF

caBIO caGRID Service Description Metadata



Server Configuration - DAISGR

```
<!-- caBIO Service Deployment -->
<service name="ogsadai/caBIODataServiceFactory" provider="Handler" style="wrapped" use="literal">
 <parameter name="ogsadai.gdsf.config.xml.file"
    value="c:/jakarta-tomcat-4.1.24/webapps/ogsa/WEB-INF/etc/caBIOConfig.xml"/>
 <parameter name="ogsadai.gdsf.registrations.xml.file"
  value="c:/jakarta-tomcat-4.1.24/webapps/ogsa/WEB-INF/etc/registrationList.xml"/>
 <parameter name="name" value="caBIO Grid Data Service Factory"/>
 <parameter name="operationProviders" value="org.globus.ogsa.impl.ogsi.FactoryProvider"/>
 <parameter name="persistent" value="true"/>
 <parameter name="instance-schemaPath" value="schema/ogsadai/gds/gds_service.wsdl"/>
 <parameter name="instance-className" value="uk.org.ogsadai.wsdl.gds.GDSPortType"/>
 <parameter name="instance-baseClassName" value="uk.org.ogsadai.service.gds.GridDataService"/>
 <parameter name="instance-operationProviders"
  value="org.globus.ogsa.impl.ogsi.NotificationSourceProvider"/>
 <parameter name="baseClassName"
  value="uk.org.ogsadai.service.gdsf.GridDataServiceFactory"/>
 <parameter name="schemaPath"
  value="schema/ogsadai/gdsf/grid_data_service_factory_service.wsdl"/>
 <parameter name="handlerClass" value="org.globus.ogsa.handlers.RPCURIProvider"/>
 <parameter name="instance-name" value="caBIO Grid Data Service"/>
 <parameter name="className" value="uk.org.ogsadai.wsdl.gdsf.GridDataServiceFactoryPortType"/>
 <parameter name="allowedMethods" value="*"/>
 <parameter name="factoryCallback"
  value="uk.org.ogsadai.service.gdsf.GridDataServiceFactoryCallback"/>
 <parameter name="activateOnStartup" value="true"/>
</service>
```

caBIO Config

```
…
<!-- caBio activities -->
<activityMap name="caBioSimpleQuery"
  implementation="uk.org.ogsadai.porttype.gds.activity.caBio.CaBioSimpleQueryActivity"
  schemaFileName="http://localhost:8080/schema/ogsadai/xsd/activities/caBIO_simple_query.xsd" />
<activityMap name="caBioQuery"
  implementation="uk.org.ogsadai.porttype.gds.activity.caBio.CaBioQueryActivity"
  schemaFileName="http://localhost:8080/schema/ogsadai/xsd/activities/caBIO_query.xsd" />
…
 <!-- caGRID specific -->
 <researchCenterInfo>
   <researchCenterByDataType>Genomic</researchCenterByDataType>
   <researchCenterName>Georgetown</researchCenterName>
   <researchCenterType>edu</researchCenterType>
   <researchCenterAddress>WashingtonDC</researchCenterAddress>
   <researchCenterPhoçe>301-xxx-xxxx</researchCenterPhoçe>
   <researchCenterFax>301-xxx-xxxx</researchCenterFax>
   <researchCenterPOCName></researchCenterPOCName>
   <researchCenterDescription>Georgetown University</researchCenterDescription>
   <researchCenterComments>Testing grid for caBIG</researchCenterComments>
 </researchCenterInfo>

   <caBioMetadata>
     <caBioSchema callback="uk.org.ogsada.porttype.gds.dataresource.CabioMetaDataExtractor"/>
   </caBioMetadata>
```

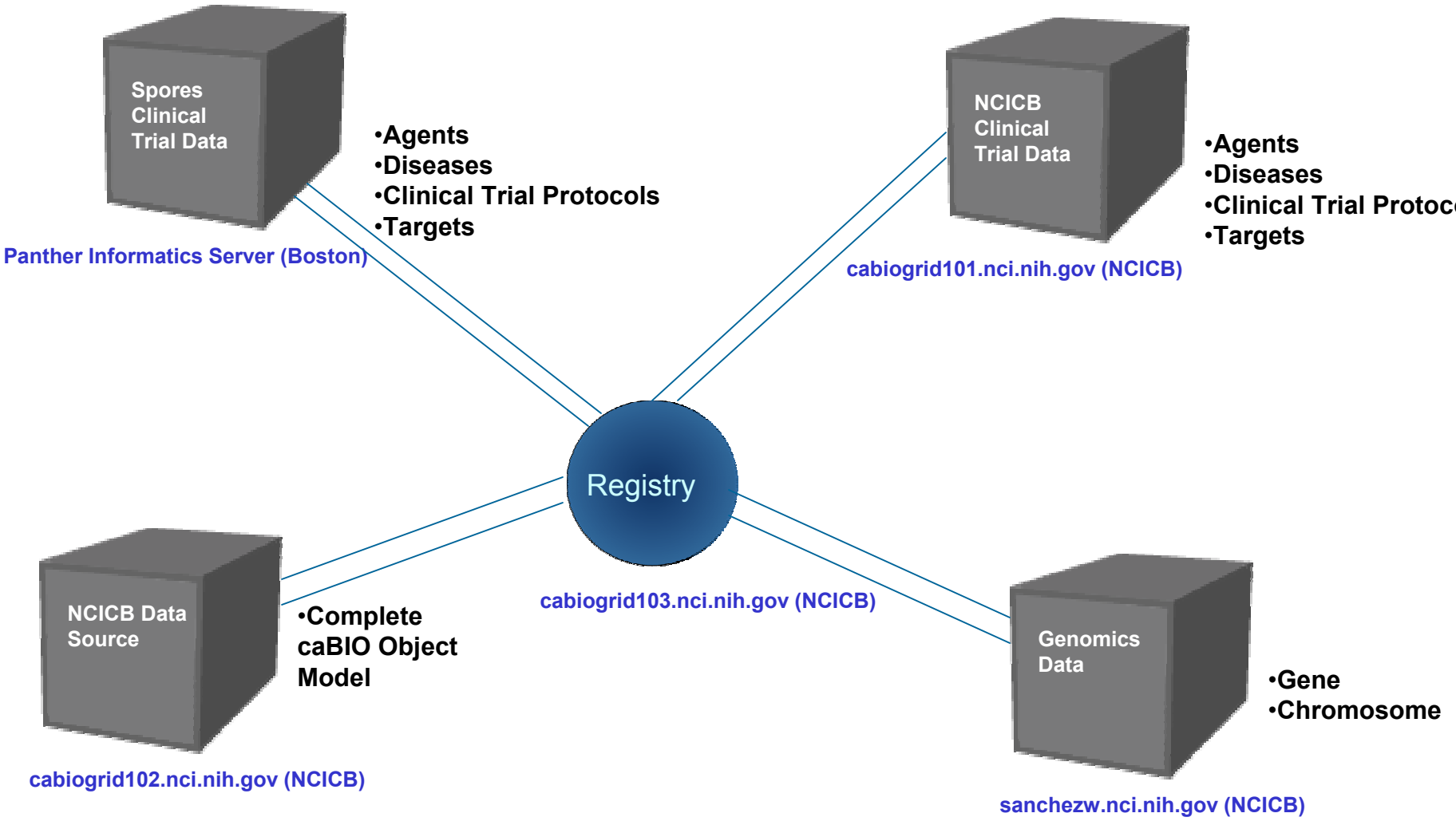
The Perform Document

```
<?xml version="1.0" encoding="UTF-8"?>

<gridDataServicePerform
  xmlns="http://ogsadai.org.uk/namespaces/2003/07/gds/types"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ogsadai.org.uk/namespaces/2003/07/gds/types
  ../../../../schema/ogsadai/xsd/activities/activities.xsd">

  <documentation>
    This example demonstrates how to parameterise an caBIO
  </documentation>

  <caBioSimpleQuery name="caBIO2.0">
        <criteria position="1" class="disease" version="1.0" id="abcd" attribute="name"
value="neoplasia"/>
        <query class="ClinicalTrialProtocol" version="1.0" id="">ClinicalTrialProtocol</query>
        <output name="Test"/>
  </caBioSimpleQuery>

</gridDataServicePerform>
```

Within the engine …

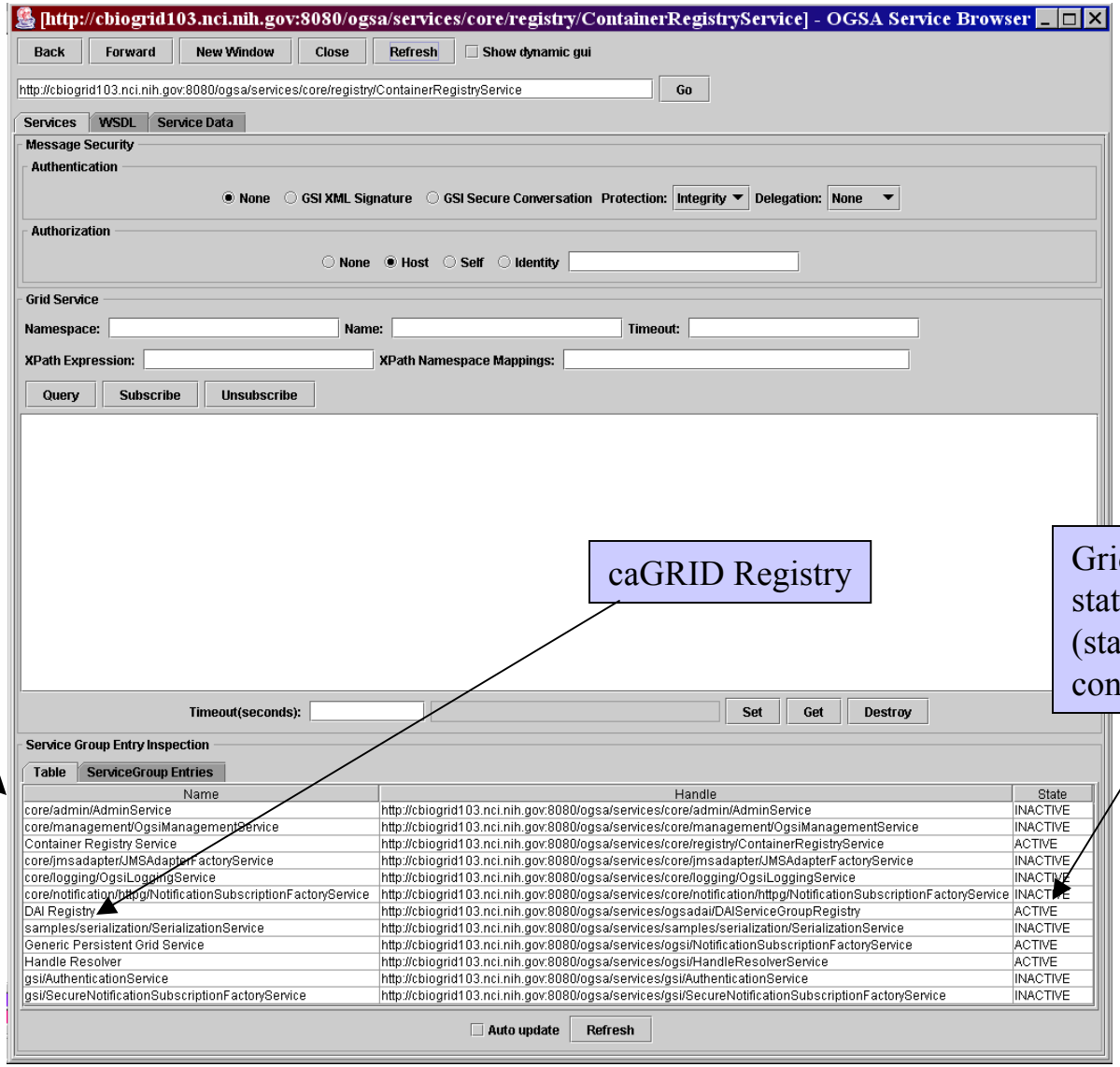
```
public void processBlock()
{

    boolean heavy =  false;
    String url = this.httpRequest;

    try {

        gov.nih.nci.common.search.SearchCriteria sc = URL2SC.map( url );
        SearchResult sr = SearchCriteria.copyDownCastSR( sc.search(), SearchResult.class );
        mOutput.put(XMLUtility.makeXMLStringDoc( sr, heavy ));

    } catch (Exception exc) {
        System.out.println("Test failed in the main of GeneDemo.java: " + exc.getMessage());
        exc.printStackTrace();
    }
    mOutput.close();
    setCompleted();
}
```

Prototype Scenario Setup



Screen Shots

caGrid – Globus GUI

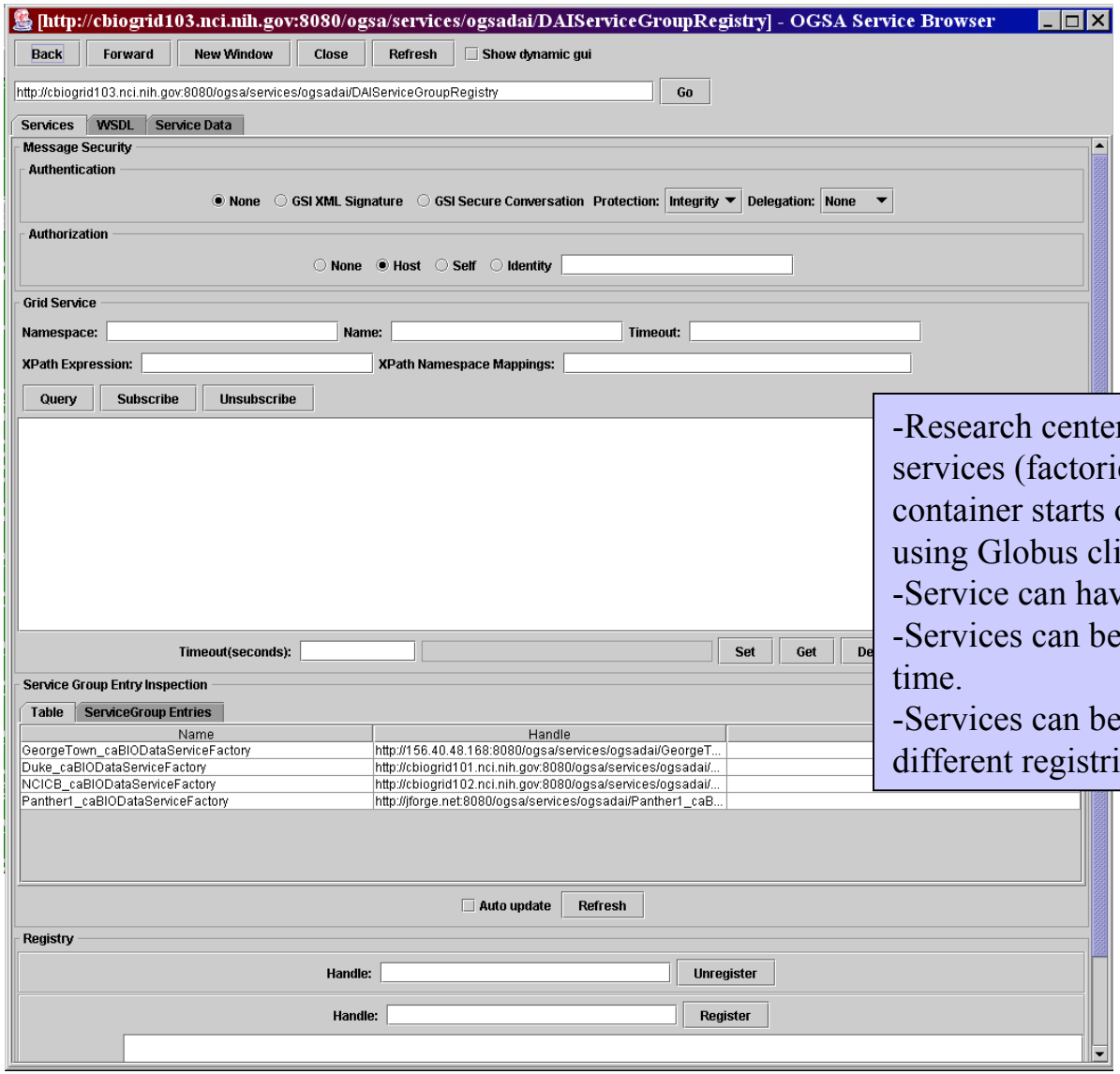


Globus Grid services list

caGRID Registry

Grid service status. Active (stateful) when container starts

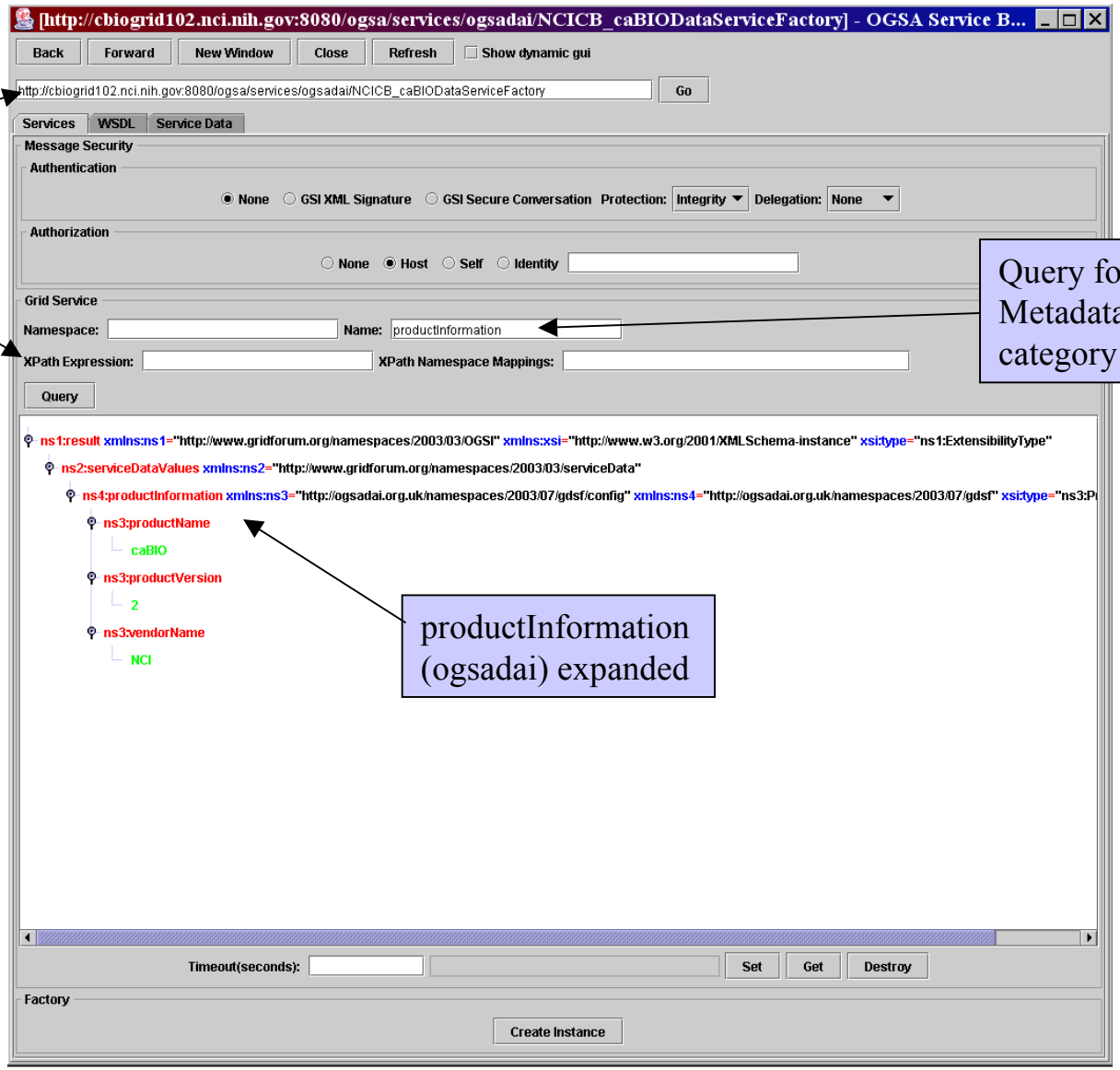
caGRID Registry



This factories represent the services advertised from Research centers

-Research centers can register services (factories) when their container starts or any time using Globus client.
-Service can have a lifetime.
-Services can be removed any time.
-Services can belong to different registries.

caGRID Factory

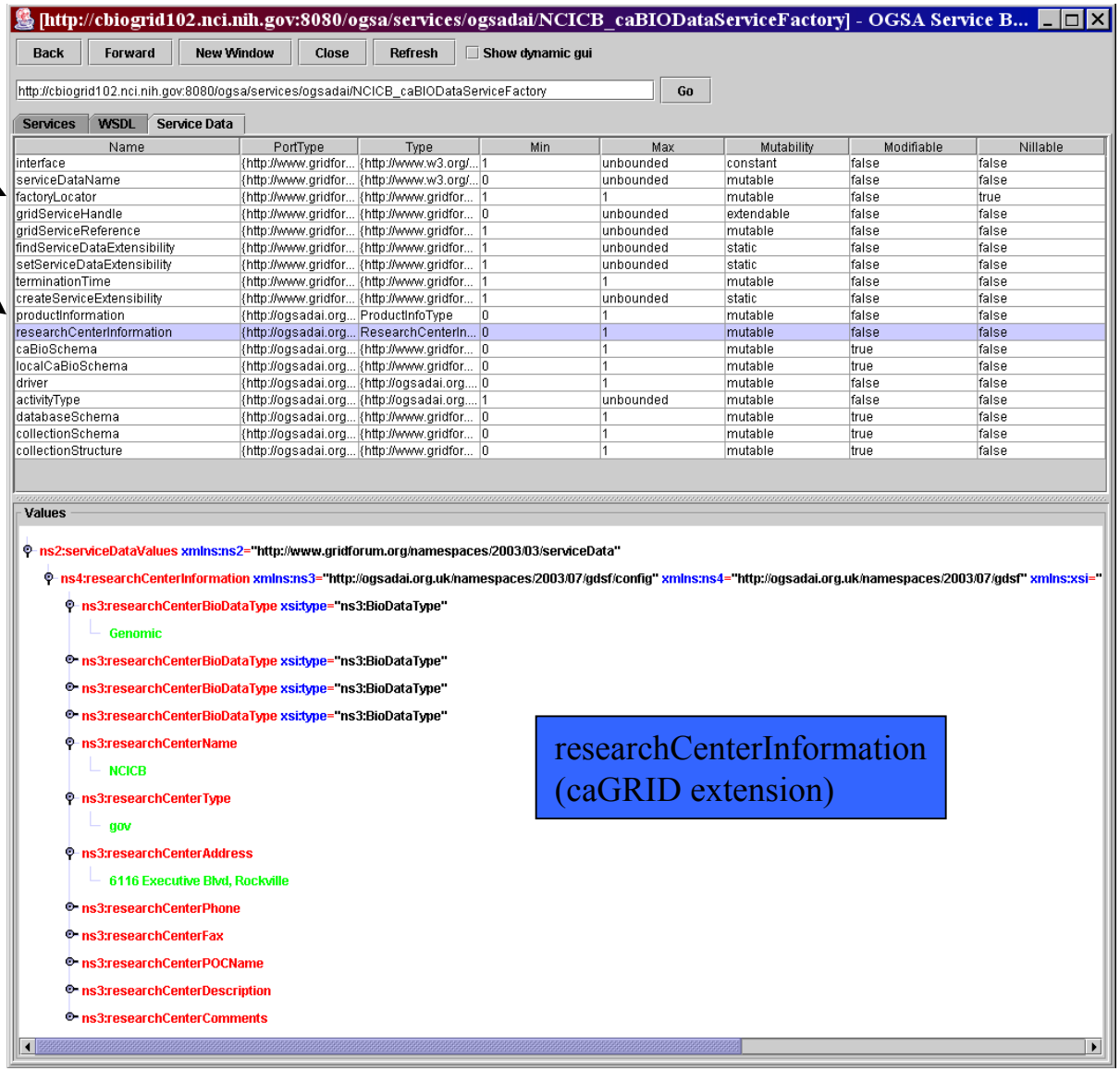


Metadata

Globus

OGSADAI

caGRID



Local caDSR

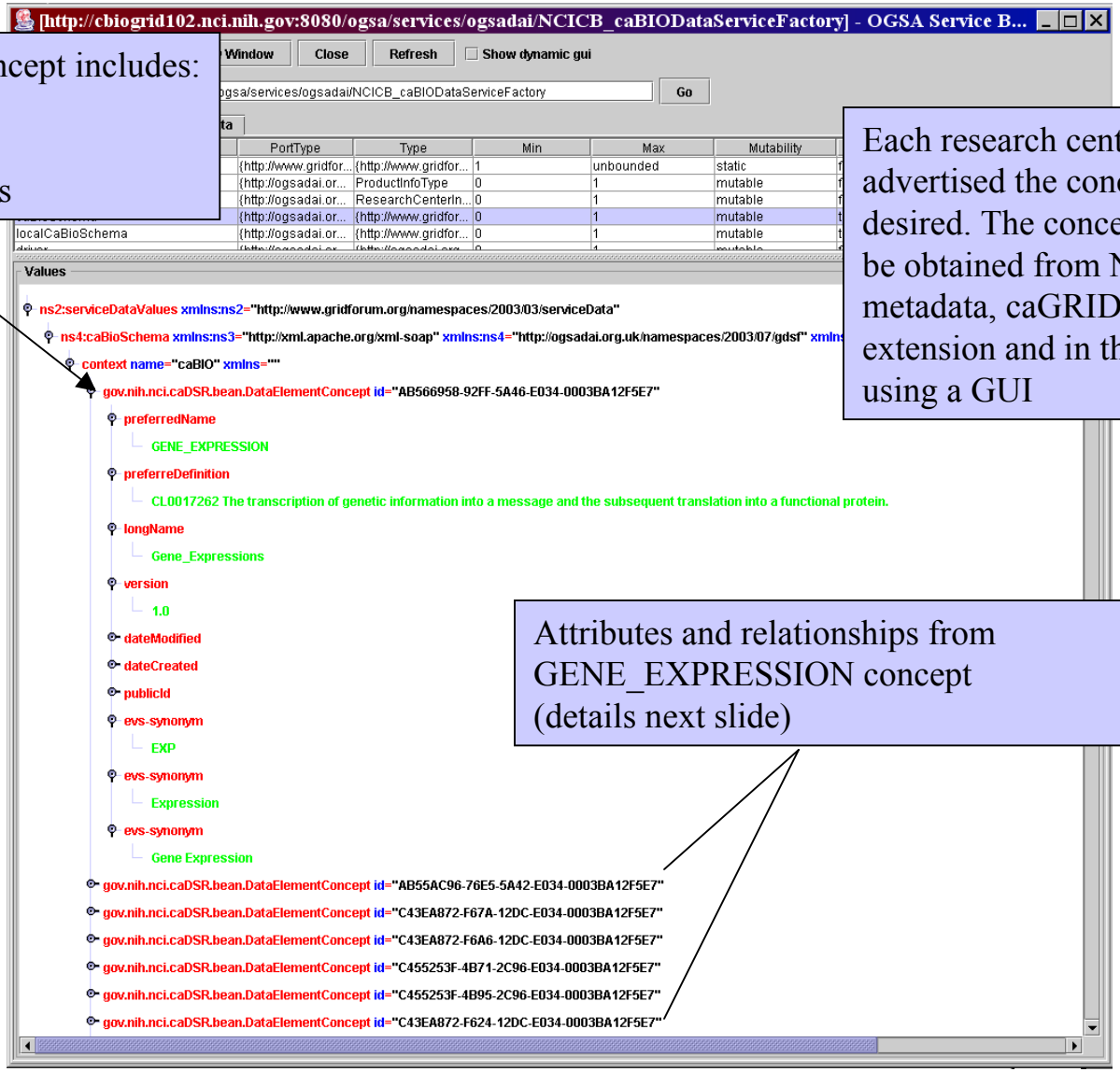
This category defines concepts advertised by a research center

List of concepts advertised

Cbiogrid102 represents NCICB grid node which advertised all concpets from caBIO object model

Metadata – GENE_EXPRESSION

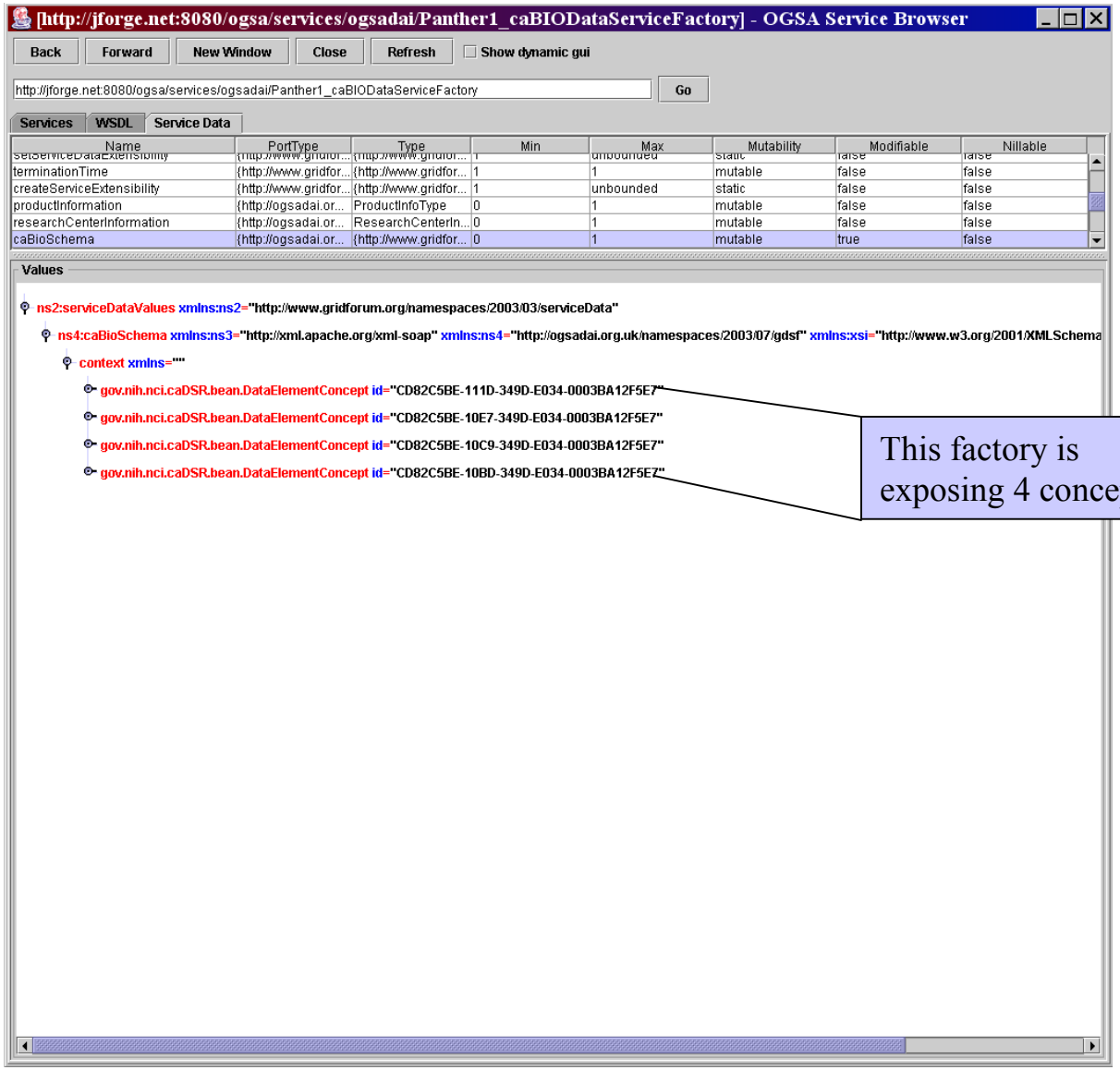
Each dataElementConcept includes:
-Concept description.
-Concept attributes
-Concept relationships



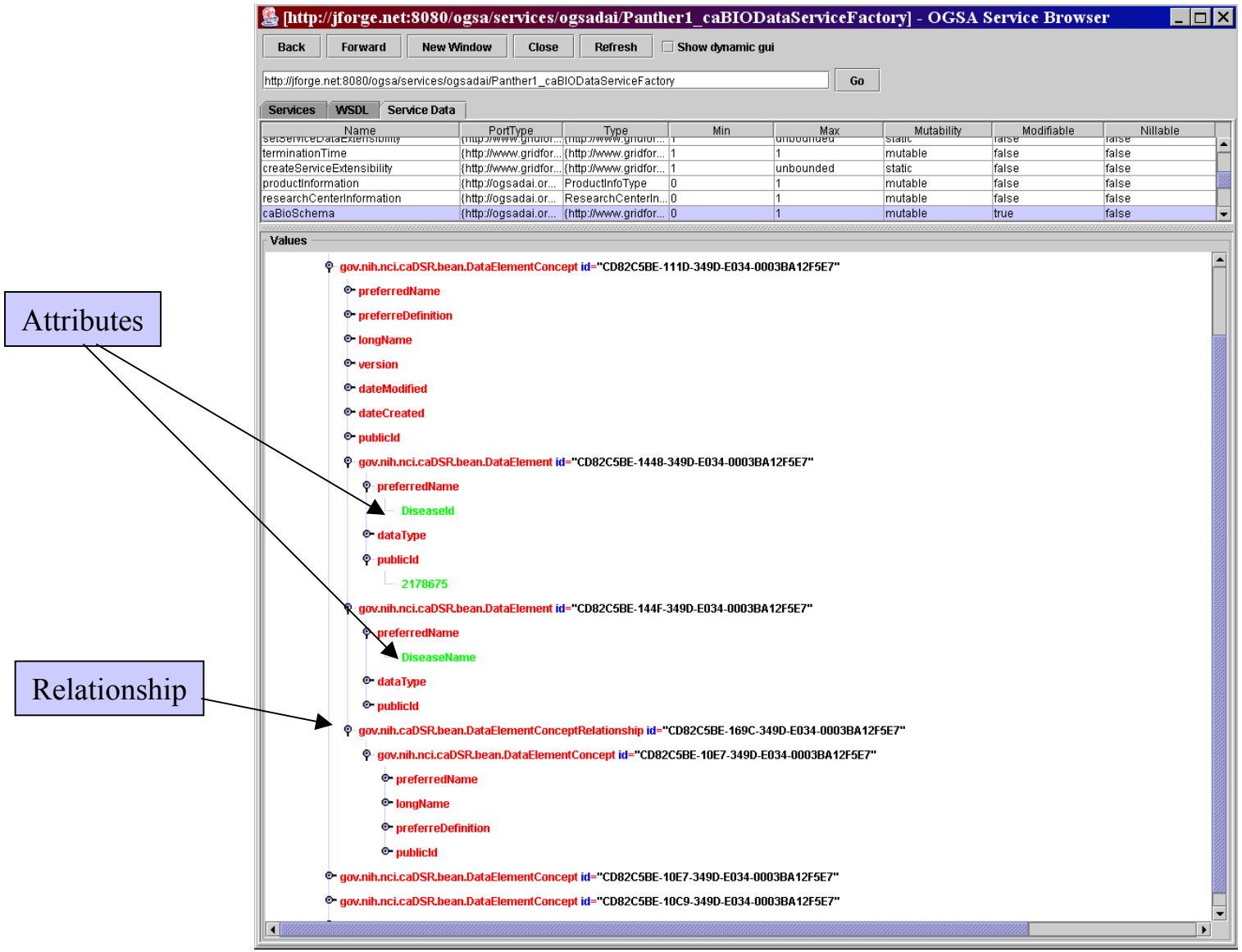
Each research center advertised the concepts desired. The concept list can be obtained from NCICB metadata, caGRID client extension and in the future using a GUI

Attributes and relationships from GENE_EXPRESSION concept (details next slide)

Another Factory



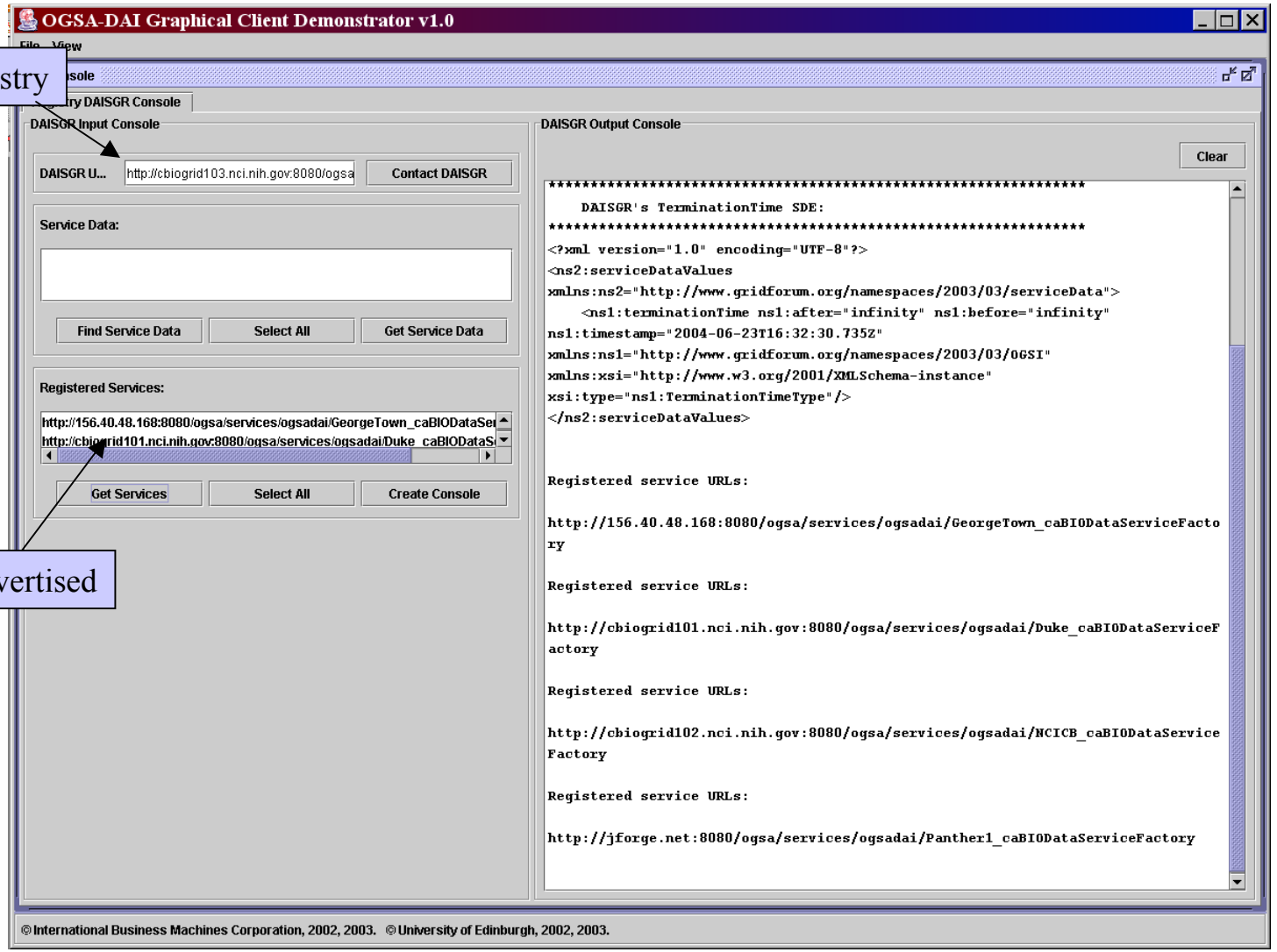
Attributes and Relationships



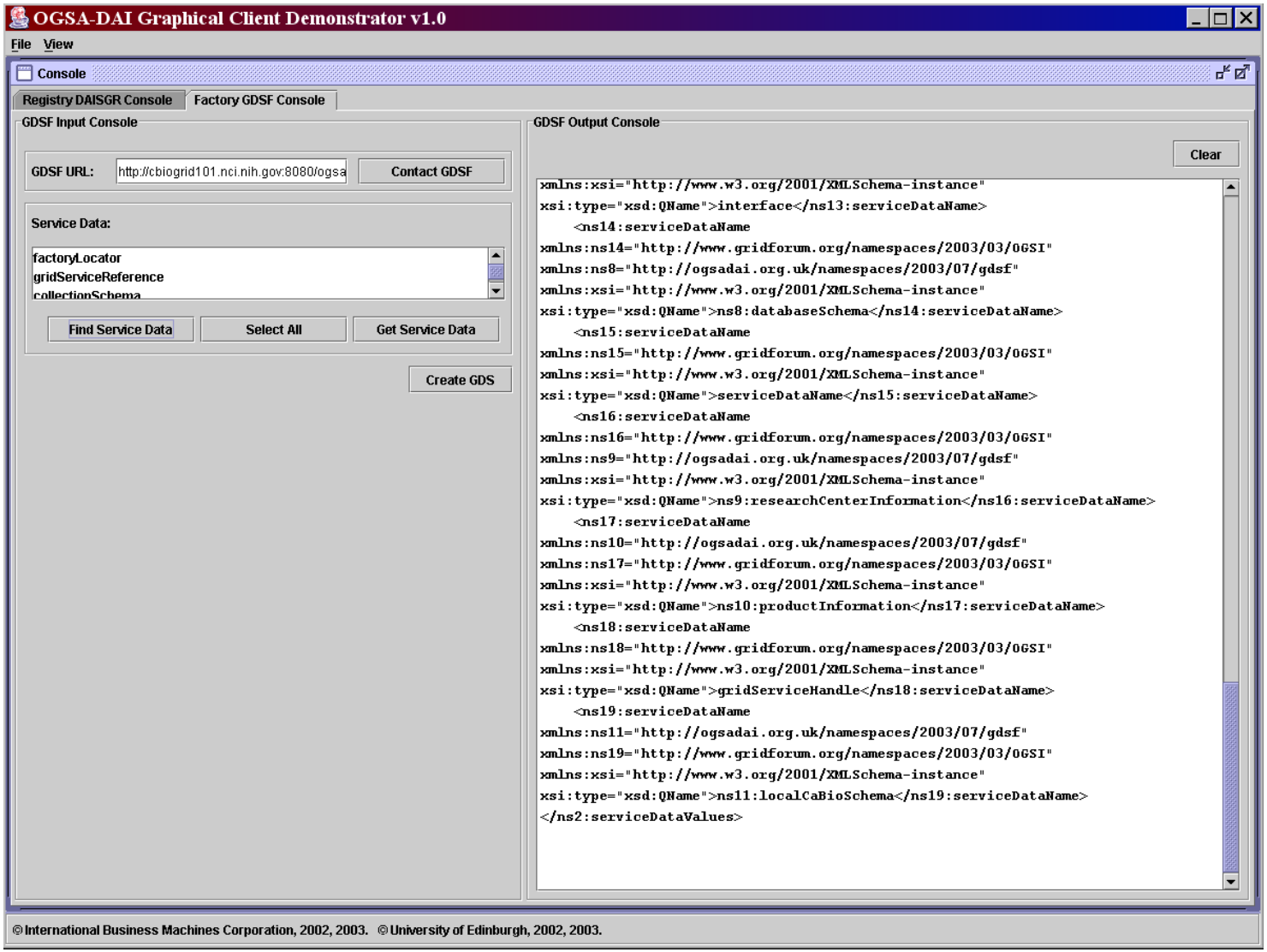
OGSA-DAI Client - Registry

caGrid Registry

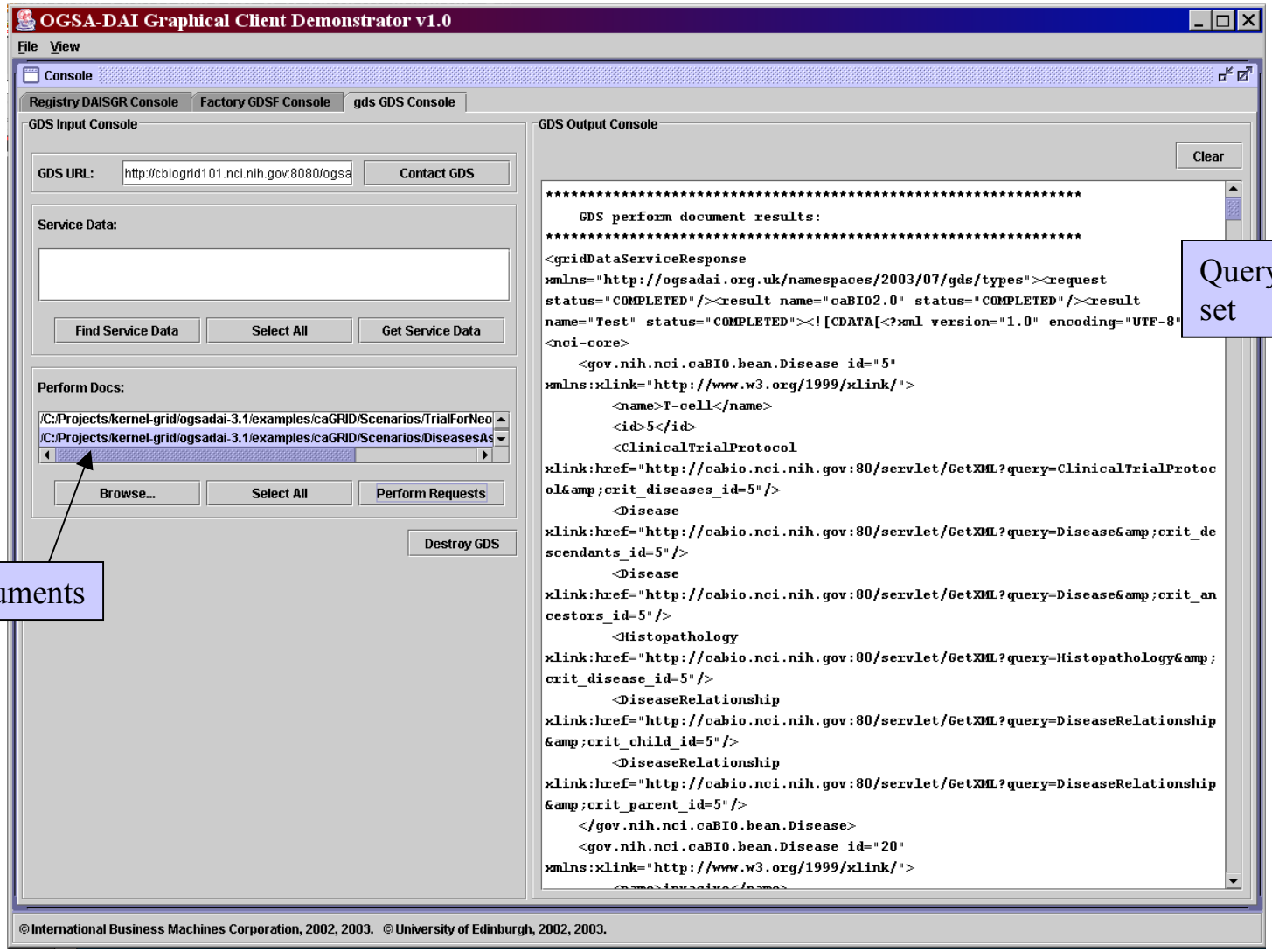
Services advertised



OGSA-DAI Client - Factory



OGSA-DAI Client - GDS



Query result set

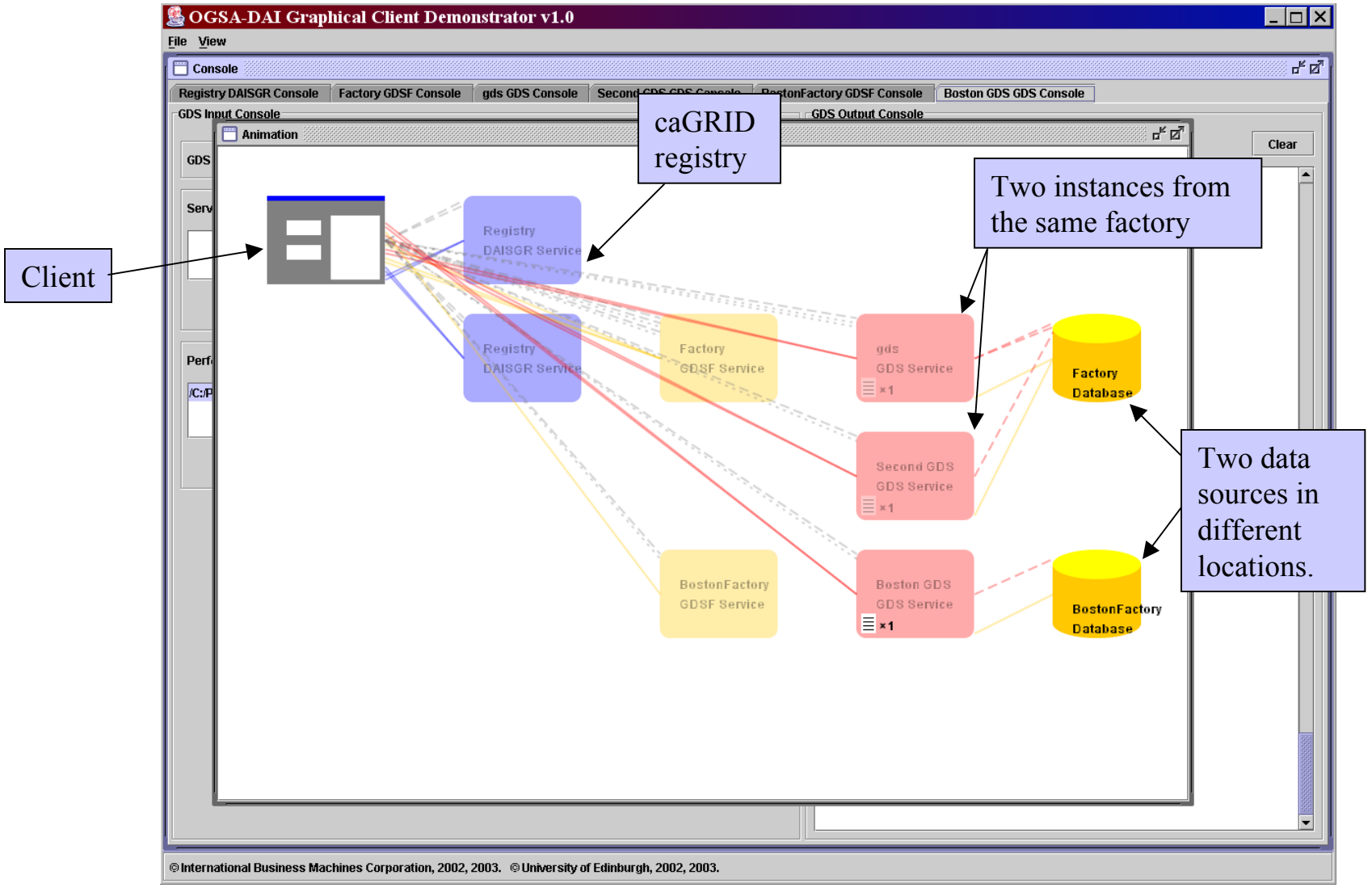
Perform documents

The Perform Document

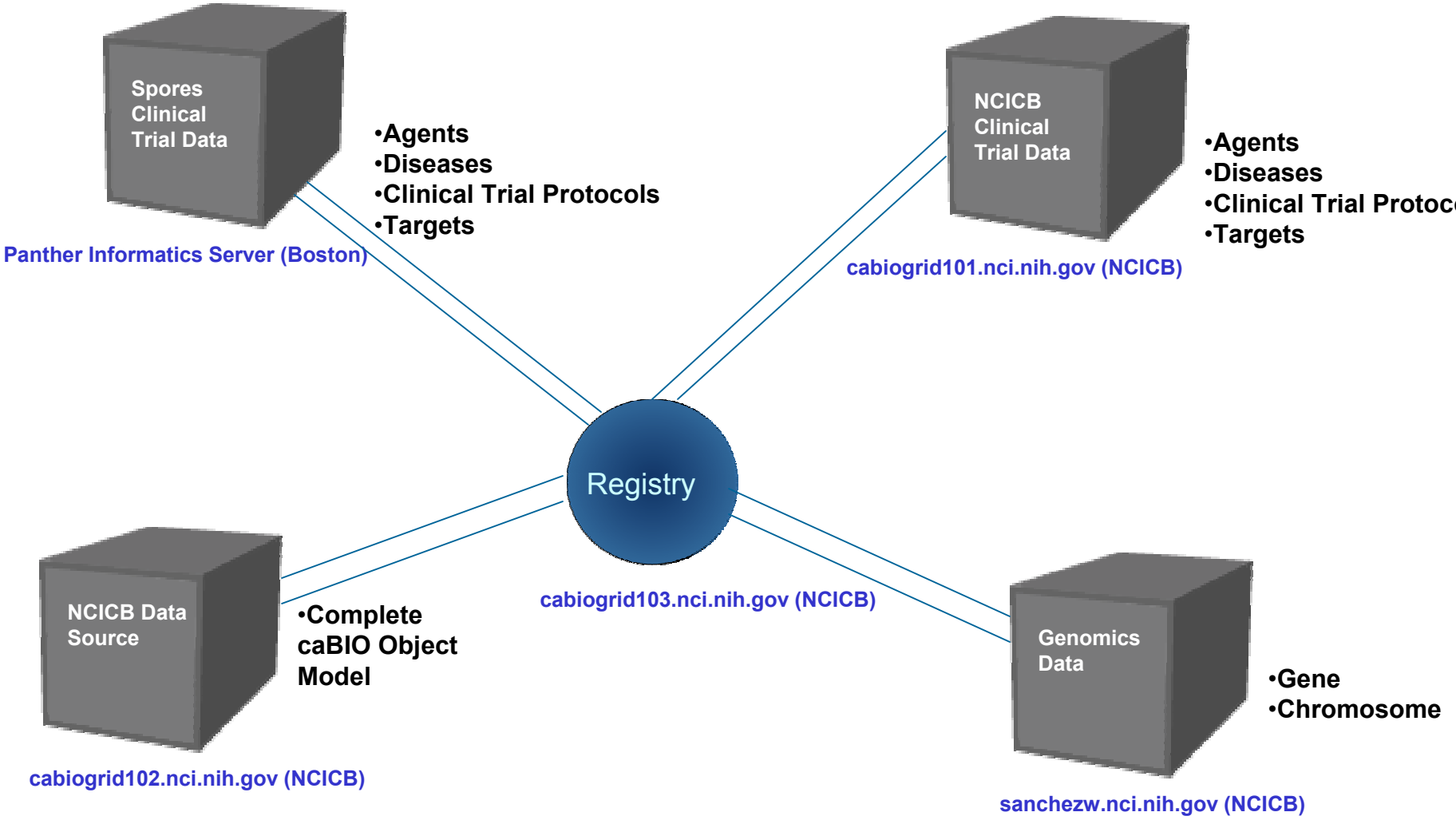
```
<?xml version="1.0" encoding="UTF-8"?>

<gridDataServicePerform
  xmlns="http://ogsadai.org.uk/namespaces/2003/07/gds/types"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ogsadai.org.uk/namespaces/2003/07/gds/types
  ../../../../schema/ogsadai/xsd/activities/activities.xsd">

  <documentation>
    This example demonstrates how to parameterise an caBIO
  </documentation>

  <caBioSimpleQuery name="caBIO2.0">
        <criteria position="1" class="disease" version="1.0" id="abcd" attribute="name"
value="neoplasia"/>
        <query class="ClinicalTrialProtocol" version="1.0" id="">ClinicalTrialProtocol</query>
        <output name="Test"/>
  </caBioSimpleQuery>

</gridDataServicePerform>
```

Show me all Trial for Neoplasia

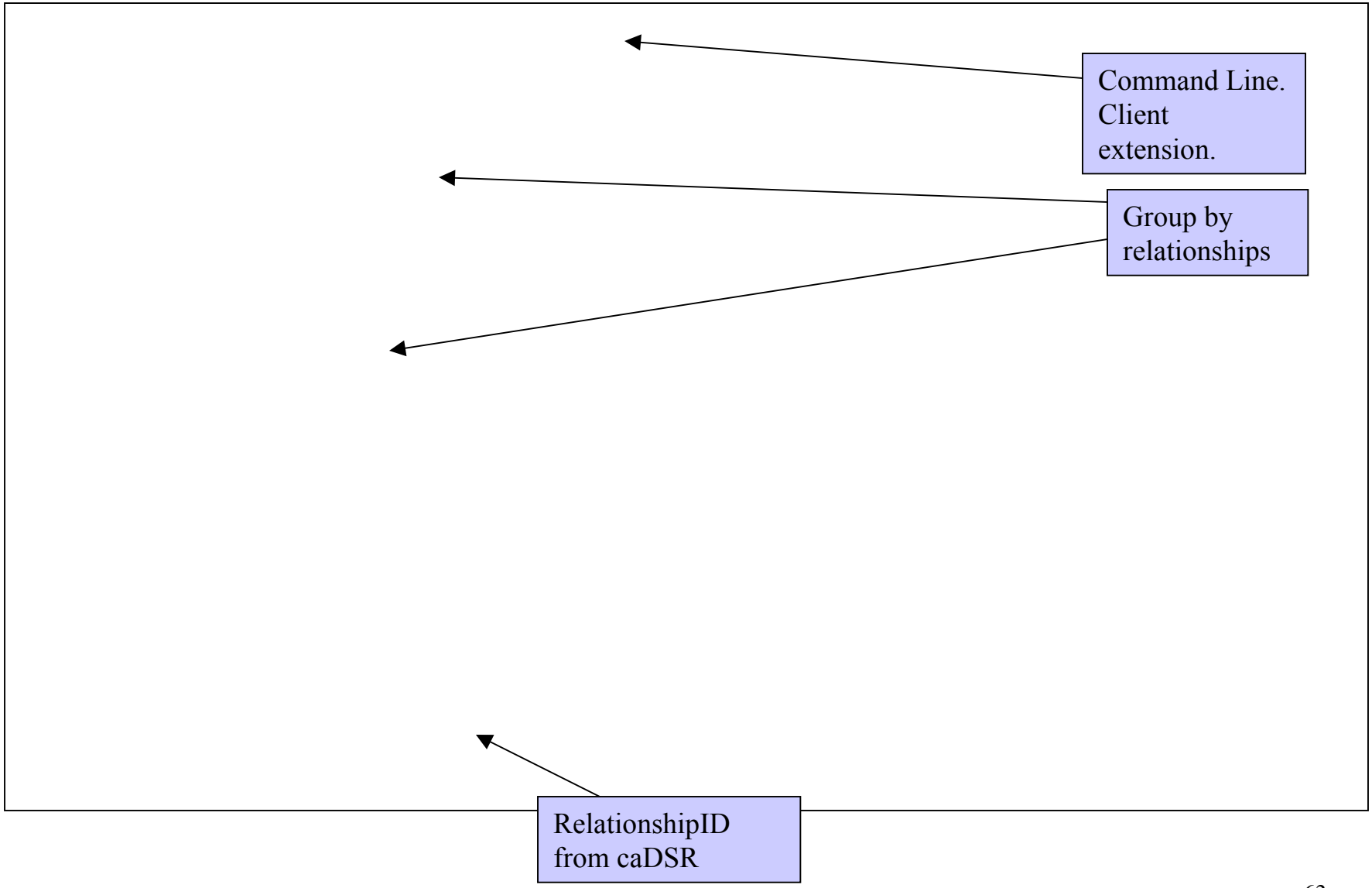
OGSA-DAI Animator – Accessing data sources



Prototype Scenario Setup – Discovery and Federated query



Discovery ClinicalTrialProtocol on the Grid



Federated query – Clinical trial protocol in Phase I